EXPLOITING SEMANTICS FROM WIDELY AVAILABLE ONTOLOGIES TO AID THE MODEL BUILDING PROCESS

A Dissertation

by

SASIN JANPUANGTONG

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Dylan Shell
Committee Members,	James Caverlee
	Matthew Fuhrmann
	Richard Furuta
Head of Department,	Scott Schaefer

December 2019

Major Subject: Computer Science

ABSTRACT

This dissertation attempts to address the changing needs of data science and analytics: making it easier to produce accurate models opening up opportunities and perspectives for novices to make sense of existing data. This work aims to incorporate semantics of data in addressing classical machine learning problems, which is one way to tame the deluge of data. The increased availability of data and the existence of easy-to-use procedures for regression and classification in commodity software allows anyone to search for correlations amongst a large set of variables with scant regard of their meaning. Consequently, people tend to use data indiscriminately, leading to the practice of data dredging. It is easy to use sophisticated tools to produce specious models, which generalize poorly and may lead to wrong conclusions. Despite much effort having been placed on advancing learning algorithms, current tools do little to shield people from using data in a semantically lax fashion. By examining the entire model building process and supplying semantic information derived from high-level knowledge in the form of an ontology, the machine can assist in exercising discretion to help the model builder avoid the pitfalls of data dredging. This work introduces a metric, called conceptual distance, to incorporate semantic information into the model building process. The conceptual distance is shown to be practically computed from large-scale existing ontologies. This metric is exploited in feature selection to enable a machine to take semantics of features into consideration when choosing them to build a model. Experiments with ontologies and real world datasets show the comparable performance of this metric in selecting a feature subset to the traditional data-driven measurements, in spite of using only labels of features, not the associated measures. Further, a new end-to-end model building process is developed by using the conceptual distance as a guideline to explore an ontological structure and retrieve relevant features automatically, making it convenient for a novice to build a semantically pertinent model. Experiments show that the proposed model building process can help a user to produce a model with performance comparable to that built by a domain expert. This work offers a tool to help the common man battle the hazard of data dredging that comes from the indiscriminate use of data.

The tool results in models with improved generalization and easy to interpret, leading to better decisions or implications.

DEDICATION

To my mother, father, and sisters.

To my wife Sirinya.

ACKNOWLEDGMENTS

This work was made possible by the help and support of my colleagues, friends, and family.

Professor Dylan Shell served as my dissertation advisor throughout my Ph.D. studies, always guiding with honor, integrity, and patience. I am truly grateful, and I appreciated the opportunity to learn from his expertise regarding academic research, professionalism, and life. I want to sincerely thank him for all of the support that he has provided over the years, especially during times of difficulty. Professors James Caverlee, Yoonsuck Choe, Matthew Fuhrmann, and Richard Furuta graciously served as the members of my dissertation committee. I want to thank each of them for their thoughtful suggestions and encouragement during my time working on this research. I also owe special thanks to Professor Paul Nelson, former dissertation committee member, for his research that inspired this work.

This research would also not have been possible without support from the Royal Thai Government and Songkhla Rajabhat University, the organizations that provided the scholarship that enabled me to complete Ph.D. work abroad. I deeply appreciate this once in a lifetime opportunity to learn how to do research and teach with confidence and professionalism. My deepest gratitude is also due to officers of the Royal Thai Government who helped me adjust to a new environment and culture.

I was very fortunate to become a member of Distributed AI and Robotics Lab, a strong research group comprised of excellent students and scholars. I would like to thank the current members of the lab (Reza Oftadeh, Yulin Zhang, Grace McFassel, Diptanil Chaudhuri, Ya-Chuan Hsu, and Reza Teshnizi) for their support and insightful feedback during the preparation of my dissertation defense. I would also like to thank former members of the lab (Ben Fine, Taahir Ahmed, YoungHo Kim, Jung-Hwan Kim, Plamen Ivanov, Changjoo Nam) for helpful feedback, suggestions, and encouragement during my research work.

Finally, I would like to thank my family who has supported me throughout all of my studies. Comfort and encouragement from them inspired me to see this work through to completion. The

most essential and influential support of all has been my wife and best friend, Sirinya Janpuangtong. She has always been by my side, sharing both happiness and sadness over the years that we have lived in the United States. This journey would not have been the same without her presence in my life, and this work is dedicated to her unwavering love and support.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supervised by a dissertation committee consisting of Professors Dylan Shell (advisor), James Caverlee, Yoonsuck Choe, and Richard Furuta of the Department of Computer Science and Engineering and Professor Matthew Fuhrmann of the Department of Political Science and Professor Paul Nelson of the Department of Nuclear Engineering.

All work conducted for the dissertation was completed by the student under the supervision of Professor Dylan Shell of the Department of Computer Science and Engineering.

Funding Sources

Graduate study was supported by the Royal Thai Government and Songkhla Rajabhat University.

NOMENCLATURE

$\mathcal{H}$	Hypothesis Space
h	Hypothesis in the Hypothesis Space
f	True Function
$\mathbf{X}$	Set of Input Features
Y	Output Feature
d	Number of Input Features
N	Number of Training Examples
p	Number of Selected Input Features
RDF	Resource Description Framework
RDFS	Resource Description Framework Schema
SKOS	Simple Knowledge Organization System
MSCA	Most Specific Common Ancestor
LASSO	Least Absolute Shrinkage and Selection Operator
RSS	Residual Sum of Squares
YAGO	Yet Another Great Ontology
k	Number of Clusters
PCA	Principal Component Analysis
HITS	Hyperlink Induced Topic Search
URL	Uniform Resource Locator
HTML	HyperText Markup Language
OWL	Web Ontology Language
MSE	Mean Squared Error

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

1. INTRODUCTION

Over the last few decades across many disciplines, we have witnessed the practical use of data to better understand and predict events of interest [2, 3, 4]. The widespread use of data analytics and data science techniques likely stems from two important factors. First, data has become available at an unprecedented rate due in part to the Internet, which has enabled individuals and organizations to make ideas and data available to a large, online audience with less expense and effort. Seemingly instantaneously, information-sensing devices (*e.g.,* mobile phones, radio-frequency identification, cameras, *etc.*), and data storage have become cheap and accessible, enabling anyone to gather heterogeneous types of data and store large quantities of data more conveniently. As a consequence, vast amounts of data in different formats and across domains have become available for building models of interesting phenomena. Second, the existence of easy-to-use commodity software for supervised learning methods makes it straightforward for anyone to analyze substantial volumes of data. Because these learning algorithms can detect subtle structures in data relatively easily without making assumptions about the process underlying and/or responsible for the data [5], they can be applied to many problems. In fact, they can be used entirely without or with limited human support. They occasionally yield interesting answers, based on the given data, sometimes before we even know precisely what question to ask [6, 7].

Democratization of various supervised learning algorithms through commodity software and the increased availability of data has caused many to question if a new era of theory-less science has began [7]. Some suggest that by advancing the learning techniques and by supplying more and more data to the algorithms, useful models will emerge. With this idea in mind, people tend to use data indiscriminately, leading to so-called data dredging [8], where correlations amongst large sets of variables are found with little regard for their meaning. As a consequence, anyone may use sophisticated tools to arrive at false conclusions. As a motivating example, Figure 1.1 demonstrates that data for cheese consumption per capita in the U.S. is highly correlated with the total revenue generated by golf courses. The relationship between these two data series may make

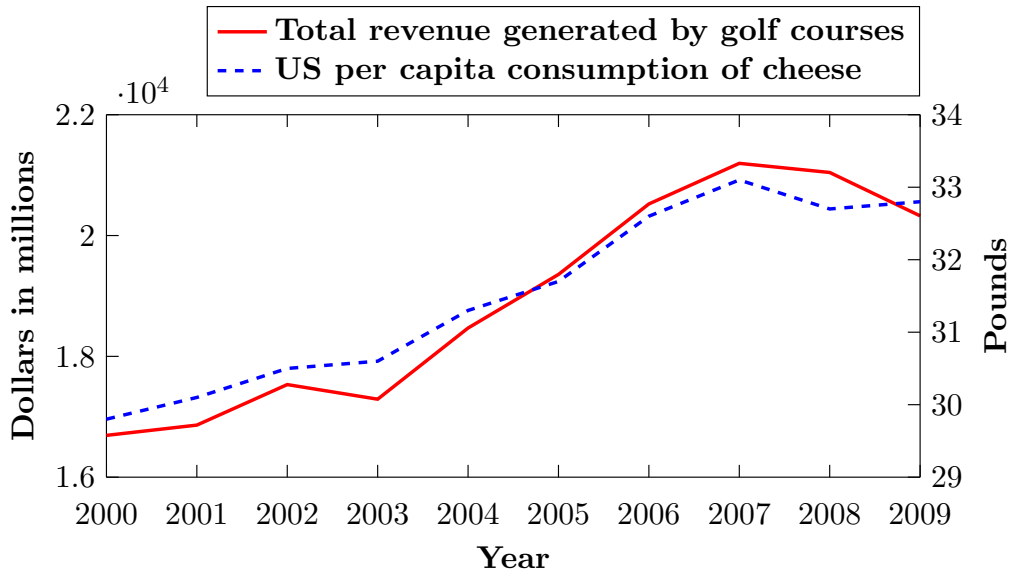

Figure 1.1: The dangers of data dredging: an example of spurious correlations within data. The plot shows per capita consumption of cheese in the US along with total revenue generated by golf courses in the US. The two are highly correlated ($\rho = 0.989$). Adapted from Spurious Correlations by Tyler Vigen, retrieved October 1, 2018, from `https://www.tylervigen.com/spurious-correlations`.

little sense to humans, as greater consumption of cheese likely has nothing to do with playing golf (nor vice-versa). But without the capacity to distinguish between genuine structure giving correlations from those that seem suspicious, learning algorithms will use cheese consumption data to predict the revenue generated by golf courses (or vice-versa) [5, 9]. With vast amounts of data becoming available, it is more difficult for machines to differentiate between signals and noise in some situations [10]. Specifically, when the amount of input data is large with respect to the number of examples, chance correlations and spurious patterns can lead to specious models, which may overfit and be difficult to interpret [11]. Academic and industrial communities focus narrowly on developing complex learning algorithms and techniques to achieve better prediction performance, but this does not get to the root of the problem: current tools do little to help prevent the use of data in a semantically loose fashion. Another amusing example used by Janowicz et. al. [12] is shown in Figure 1.2. It illustrates the importance of data semantics. We can see that all three quantities on the road sign were added together without considering their meanings yielding

Figure 1.2: A sign in New Cuyama, Santa Barbara County, California, reprinted from New Cuyama, California, in Wikipedia, retrieved October 1, 2018 from `https://en.wikipedia.org/wiki/New_Cuyama,_California`. Copyright 2007 by MikeGogulski.

a new quantity with label "TOTAL" that has no significance.

This dissertation broadens the development, examining the entire model building process rather than focusing on the learning step so that semantics may be effectively incorporated in the process. This work interprets data not merely as a vector of values by associates it with a mental concept that attach meaning to the data. Reasoning about the relationship between the concepts is carried out to assess the relationship between the data that are associated with the concepts. That is, the decision of whether or not to use particular data in a learning problem is elevated to an abstract, conceptual level rather than deciding purely at the data level. Consequently, models are built with semantically pertinent data rather than through mere correlations or patterns within data.

This work benefits from rich semantic information present in structures of existing knowledge resources called ontologies [13]. A quantitative notion of relationship between data at the conceptual level is defined through an ontological structure so that data semantics are considered in determining their relationships. We introduce a metric, called conceptual distance, computed from an ontological structure to incorporate semantic information into the model building process. The effectiveness and utility of this measurement is demonstrated through the performance of the model built. Exploiting existing ontologies can relieve the burden of creating our own semantic structures, which would be a time consuming and labor-intensive process. Moreover, these ontologies are usually equipped with several convenient services that work as interfaces to interact with such ontologies. One important service is a SPARQL endpoint, which enables the computation of the conceptual distance to be accomplished without performing any pre-processing steps or loading the entire ontology to a local computer.

1.1 A Preview of the Approach

The overarching hypothesis of this work is that semantic information, as codified in an ontology, can be usefully brought to bear in the model building process. Semantic information derived from an ontology is used to help exercise caution in using data to build a model. In particular, this work focuses on exploiting this information in feature selection. Feature selection has become a vital component in machine learning processes, especially in the presence of hundreds or thousands

of input features. It has been shown in both theory and practice to enhance learning efficiency, increase predictive accuracy, and reduce complexity of the learned results [14, 15]. In general, the goal of feature selection is to choose the *best* feature subset among all available subsets [16, 17]. Therefore, the main component in feature selection is the computation of evaluation criteria for assessing the goodness of individual features or the whole feature subset. This work defines the evaluation criteria of feature relevance and redundancy from the conceptual distance computed from an ontology rather than using traditional measurements at the data level. Evaluation in this way enables the semantics of the features to be considered in selecting and/or discarding input features. The conceptual distance is also adopted as a guideline for systematically exploring an ontological structure in order to automatically recommend relevant features. This feature recommendation procedure can help novices to build models with similar predictive performance to those produced by domain experts.

1.2 Summary of Contributions

Launching from the overarching hypothesis, this dissertation makes the following contributions:

- A novel means for evaluating the relationship between the features is introduced. The notion of relatedness between features at the conceptual level is defined in the form of a metric, called conceptual distance, computed over a certain ontological structure to inform the relationship between the features at the data level.
  - Simple and reusable SPARQL queries are developed to compute the conceptual distance on different large-scale ontologies, appropriated even when local resources are limited.
  - This metric is shown to be able to compute effectively across ontologies
- The conceptual distance becomes the foundation of this dissertation, where it is used as a tool for developing a feature selection method that can handle both feature relevance and redundancy.

  - Features' labels are the only input used by this method to determine relevance and redundancy of features via the conceptual distance without using features' data.
  - The effectiveness and informativeness of the conceptual distance in feature selection are examined through several experiments, showing that this method perform comparable to or even better than the tradition data-driven methods in some cases.

- An end-to-end model building framework is developed. This framework incorporates rich semantic information from an ontology and requires minimum input from a user.
  - A feature recommendation is introduced, using the conceptual distance as a guideline to explore an ontological structure and retrieve relevant input features for building a model.
  - This framework is shown to help a novice to produce a model with similar performance to that built by domain experts, illustrating the effectiveness of the conceptual distance in retrieving useful and sufficient information to build a good model.

1.3 Organization of this Dissertation

This dissertation is organized as follows: background about information extraction and supervised learning along with existing works that make use of knowledge and semantics to improve the performance of learning algorithms are discussed in Chapter 2. The conceptual distance computed from an ontology accompanied by the SPARQL queries for large-scale ontologies are described in Chapter 3. In Chapter 4, we detail our feature selection method that makes use of the conceptual distance to evaluate feature relevance and redundancy. The implementation of the method and the experimental results demonstrating the usefulness and effectiveness of the method are also provided in this chapter. The proposed end-to-end model building process along with the details of its components are discussed in Chapter 5. Finally, the conclusion and future work possibilities are addressed in Chapter 6.

2. BACKGROUND AND RELATED WORK

This work is an attempt to bridge the gap between information extraction and supervised learning. Generally speaking, both disciplines aims to discover structures from unstructured data. While the former task focuses on extracting structured knowledge and representing them in the form that a machine can understand and reason with, the latter task concerns in detecting structures that are useful for predicting or explaining a phenomenon of interest. By exploiting semantic information codified in structural knowledge produced by information extraction to supervised learning, it can aid the latter task in a number of ways, including in improving the performance of a model being built. This section provides some background of these two disciplines along with existing works that attempt to connect them together.

2.1 Supervised Learning

Supervised learning generally assumes that there is an unknown function *f*, which is drawn from a hypothesis space $\mathcal{H}$. Given a set of examples, called a training set, assumed to be generated from *f*, supervised learning learn from these training examples a function $h \in \mathcal{H}$ that approximates *f* [18]. Typically, the hypothesis space $\mathcal{H}$ is very large (*e.g.,* the set of all polynomials), and the given training examples are the only source of information to help pinpoint the correct (or a good enough) hypothesis from this space. However the training set usually contains only a small subset of the entire set of observations [18, 19], it provides what the output should be for only a small percentage of the cases. That is, the training set by itself is not sufficient to find the unique solution. This is an instance of an ill-posed problem [20].

2.1.1 Inductive Bias of Learning

In order to obtain a unique solution with limited amount of data, extra assumptions need to be made to place additional constraints on $\mathcal{H}$ or allow a unique hypothesis to be selected from many plausible ones. The set of such assumptions is called the inductive bias of the learning algorithm [20]. One example of this type of bias is when a certain class for $\mathcal{H}$ is assumed (*e.g.,*

assuming a linear function resulting in constraining $\mathcal{H}$ to a subspace containing all linear functions). Among all hypotheses in $\mathcal{H}$, choosing the hypothesis that minimizes prediction error on the training examples is another inductive bias.

2.1.2 Generalization of a Hypothesis

The goal of machine learning is not merely to find a hypothesis h that minimizes the training error, but one that would generate the correct output for unseen examples outside the given training set. How well a hypothesis trained on the training set predicts the right output for new examples is called generalization [18, 21]. An important source of information that can help a learning algorithm to find a hypothesis that generalizes beyond the given training set is the given $\mathcal{H}$. More precisely, we can find $h \in \mathcal{H}$ that has the minimum training error but if $\mathcal{H}$ is not chosen well, no matter which $h \in \mathcal{H}$ we pick, we will not have good generalization [20, 18]. In order to obtain the best generalization, the complexity of $\mathcal{H}$ should match with the the complexity of f. If the selected $\mathcal{H}$ is less complex than f, it does not have enough capacity to include f; this is underfitting (*e.g.,* when trying to fit a line to data sampled from a third-order polynomial). On the other hand, if $\mathcal{H}$ is more complex than f, an over-complex hypothesis may capture not only the underlying function but also noise in the data; this is overfitting (*e.g.,* when fitting a sixth-order polynomial to noisy data sampled from a third-order polynomial).

2.1.3 Model Selection

The problem of choosing among possible $\mathcal{H}$ is called model selection [22]. Since f is unknown, heuristics are usually exploited in selecting an appropriate $\mathcal{H}$. If we have a large training set, we may choose a complex $\mathcal{H}$ to avoid underfitting. As the training set size increases, the variability of examples decreases, so that the complexity of a hypothesis from $\mathcal{H}$ can be constrained with the data. If the training set is small, the risk of overfitting is the main concern. Therefore, a simpler $\mathcal{H}$ is preferable [20, 18]. Even though a simpler hypothesis is constrained more, it is less affected by single examples so that it would generalize better than a complex one.

2.1.4 Using Knowledge to Enhance Supervised Learning

Supervised learning is an ill-posed problem, as a set of training examples alone does not determine a unique solution. Extra information is required to yield a good solution from a large hypothesis space. Different approaches have been developed by exploiting various forms of knowledge as a source of extra information to enhance the performance of supervised learning algorithms. The following sections survey these approaches.

2.1.4.1 Declarative Bias

Several existing works formally use background knowledge from a knowledge base to alter the search performed by learning algorithms. Declarative bias is determined from the knowledge base and then introduced into the learning process to guide the search for the best hypotheses. Different forms of knowledge have been used to derive this bias for different learning tasks. Russell and Grosof [23] proposed learners that make use of background knowledge represented by monotonic and non-monotonic logic to automatically learn a single concept. Declarative bias has also been used in the equation discovery task [24]. The LAGRAMGE system was developed by using declarative bias in the form of context free grammars to help limit the search of potential equations.

2.1.4.2 Causal Modeling

Several learning algorithms, such as Knowledge-Based Artificial Neural Network [25] and Bayesian beliefs networks [26], employ background knowledge to form an initial model and then use data to validate that model. Even though these algorithms have been demonstrated to outperform purely inductive learning [18], their main limitation is that they can only accommodate a specific knowledge representation and a learning method. Causal modeling is another approach that develops a model, called causal model, to describe a system of interest by specifying causal relationships between variables [27]. One example of this approach is Structural Equation Modeling [28], which includes a diverse set of mathematical models, computer algorithms, and statistical methods that fit networks of constructs to data. Constructing such a causal model accurately requires knowledge from domain experts. Our work aims to reduce the necessity of domain experts

in the model building process by exploiting high-level knowledge in the form of an ontology so that anyone could benefit from our approach.

2.1.4.3 Exploiting Knowledge in Feature Selection

Some works exploit knowledge to guide the process of selecting features for building a model. The source of knowledge varies from domain experts, relevant publications, or relevant datasets [29]. Zhao et al. [30] integrate information from different data sources as knowledge to extract an intrinsic global geometric pattern for gene selection. Ben Brahim and Limam [31] use prior knowledge obtained from domain experts and relevant publications to develop three different feature selection procedures. Groves [32] develops a framework that asks for knowledge presented in a specific form from a user to systematically guide feature selection. Lee et al. [33] make an assumption that features themselves have meta-features that are predictive of their relevance to a certain task, and model their relevance as a function of the meta-features using hyperparameters, called meta-priors. The general idea of this dissertation is similar to that of this work, but here concepts from an ontology are used as a sort of meta-feature and the conceptual distance computed from the ontology is used to determine relevance of the corresponding features.

2.2 Information Extraction

Information extraction automatically extracts structured information from unstructured (*e.g.,* web documents, collection of books) or semi-structured (*e.g.,* tables, lists) data sources so that a machine can semantically interpret and automatically make use of those data [34]. This task generally concerns analyzing human language texts using techniques from natural language processing. It relies on hand-crafted extraction rules or hand-tagged training examples to extract relations between entities from input texts. The main goal of information extraction is to construct a source of structural knowledge so that machines can utilize such knowledge to solve complex tasks [35, 36].

2.2.1 Ontology Learning

Since building ontologies manually is extremely labor-intensive and time-consuming, information extraction is applied to construct an ontology from existing unstructured data sources such as

web documents. This process involves extracting the corresponding symbols within a domain and relationships between the concepts that these symbols represent from a corpus of natural language text, and encoding them with an ontology language (*e.g.,* Web Ontology Language, Resource Description Framework) [37, 38]. One example of an ontology that is constructed by information extraction is DBpedia [39], in which Wikipedia pages are used as a source to extract structural knowledge. Another example is YAGO [40], which combines structured information extracted from Wikipedia with data from WordNet [41]. These general-purpose ontologies cover a wide range of domains such as geographic information, people, companies, genes, music, *etc.* Not all ontologies are the products of information extraction. CYC [42] is one such example as it is constructed manually by domain experts.

2.2.2 Ontology-Based Applications

Since many ontologies have become accessible in recent years, many applications have adopted these ontologies as a source of high-level knowledge and/or semantic information to enhance performance of the applications. The following sections provide examples of these applications in different disciplines.

2.2.2.1 Bioinformatics

Semantic information and structural knowledge in ontologies have been widely exploited in biological and biomedical researches [43]. These disciplines make use of ontologies in identifying important knowledge: drug-drug interaction [44], candidate genes for diseases [45]. Some works [46, 47] define metrics over ontological structure to determine a relationship between biological elements.

2.2.2.2 Geographic Information Systems

Vocabularies, as described in an ontology, is used to integrate geographic information from different sources. Fonseca et al. [48] develop an ontology-driven geographic information system that acts as a system integrator, enabling users to browse through different geographic information sources. Hwang et al. [49] use knowledge in an ontology to enhance searching on maps, taking

preferences of users into account.

2.2.2.3 Feature Selection

Jeong and Myaeng [50] use the taxonomic hierarchy in Wordnet to select features specifically for event recognition and type classification. Paulheim [51] developed a general learning system that can automatically augmented a given dataset with additional features retrieved from linked data. His system looks for features corresponding to a given set of entities and then determines their relevance via correlations within the data. But the limited data published in the linked data form restricts the sets of entities and types of features that this approach can work with.

2.3 Semantic Information

One form of semantic information that is useful across many tasks involving word meanings is an association between words. Assessing association between words has largely proceeded by using contextual information obtained from a corpus of documents. Different techniques have been developed to obtain this information. Latent Semantic Analysis applies statistical methods to a large corpus of text to extract and represent a set of concepts related to words and documents that contain those words [52]. The hypothesis underlying this approach is that linguistic items with similar distributions have similar meanings [53]. Another approach is word embedding, such as *word2vec* [54], where words are represented by vectors. Intuitively, this approach involves an embedding from a space with many dimensions per word to a continuous vector space of much lower dimension. A corresponding vector of a word is usually constructed from neural networks, co-occurrence matrices, *etc.* [55]. Another source of semantic information is structured knowledge. A lexical database, WordNet [41], is an example of such source, in which semantic relations between words are expressed in this database. This dissertation exploits this source of semantic information, as codified in the form of an ontology, explicitly describing the relationships between ontological concepts.

2.3.1 Exploiting Semantic Information in Feature Selection

Several works have taken semantics of features into consideration when selecting relevant features for building a model. Lampos et al. [56] apply neural work embeddings on a large collection of search queries, and then use the embeddings in conjunction with conventional feature selection methods to encourage a level of topicality in the selected features. The approach is shown to be effective in text regression for the task of flu surveillance. Structural knowledge in an existing knowledge base is also used in feature selection. Chua and Kulathuramaiyer [57] employs noun synonyms and word senses expressed in WordNet to select features for text categorization. The approach presented in this dissertation does not use semantic information at the textual level, but consider the relationship between features at an abstract, conceptual level, as codified in the structure of an ontology. Therefore, this approach does not limit to a specific learning problem, where texts are used as features for building a model.

2.4 Summary

This section provides a background overview of supervised learning and information extraction. Existing works that incorporate knowledge to enhance the performance of supervised learning are provided. Details of engineering ontologies automatically via information extraction along with several ontological-based applications are discussed. Finally, two sources of semantic information are described accompanied by feature selection approaches that take semantic of features into consideration. In the chapter that follows, we will describe how to derive semantic information from a particular ontological structure along with how to use this semantic information to enhance feature selection and feature recommendation.

3. THEORY AND APPROACH

This chapter describes a semantic information that can be derived from an ontological structure and the idea of how to bring this information to bear in a model building process. The notion of relatedness between ontological categories is defined by using this semantic information and is expressed in terms of a metric, which is called conceptual distance. This distance is used to evaluate a relationship between real world entities semantically. The reusable queries in SPARQL language is developed for retrieving the semantic information and computing the distance on the fly in an ontology independent fashion.

3.1 The Nexus between the Notions of Relevance at the Conceptual and Data Levels

The traditional way to examine the relationship between entities is by using their corresponding data. Some entities are physical (*e.g.,* humans, tables, shopping malls, *etc.*), and some are essentially mental (*e.g.,* literacy, depression, obesity, *etc.*). One entity is determined to be related to another entity if their data explain the variance of each other. We call this the *'relevance at the data level'*. This notion of relevance is realized by data-driven measurements, such as correlation coefficient or mutual information. In this work, the relationship between entities is determined at the conceptual level, which informs the relationship at the data level. The notion of similarity between entities is used as the notion of the *'relevance at the conceptual level*. The hypothesis is that with little similarity, the relationship between two entities observed at the data level may occur by chance or via noise. On the other hand, if the two entities are very similar, the relevance occurring between these entities at the data level is more plausible. The connection between the two notions of relevance is shown in Figure 3.1. Using this connection, a discretion is exercised by a machine through the notion of similarity between entities at the conceptual level to help a model builder avoid the pitfalls of data dredging. That is by placing more trust on the information obtained at the conceptual level than the information obtained at the data level, some decisions in the model building process is lifted to make at the conceptual level rather than at the data level. In the sec-

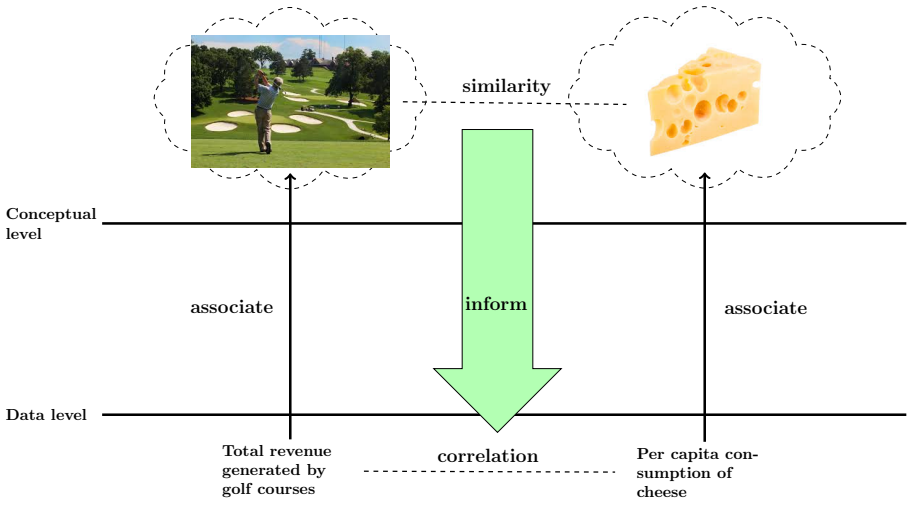

Figure 3.1: This figure shows the nexus between the notions of relevance at the conceptual and the data levels, using the correlation between total revenue by golf courses and per capita consumption of cheese as an example. This work associates each vector of data with a metal concept corresponding to an entity so that relationship between corresponding entities can be determined via the notion of similarity. The relationship at the conceptual level then informs the plausibility of the relationship observed at the data level.

tions that follow, we describe an ontological structure that carries useful information for assessing similarity between entities and how to derive the similarity measurement from this structure.

3.2 Category Organization

A category or class organization of entities is a general structure available across ontologies. An information that can be derived from this organization is how properties are shared among categories, determining their similarity. This notion of similarity between categories is used to estimate the similarity between entities that belong to the categories. There are two main components in this organization: a set of ontological categories and an ontological relation between categories.

3.2.1 An Ontological Category

An ontological category or an ontological class represents a set of entities sharing some common properties. More precisely, an ontological category is equivalent to a unary predicate in first-order logic. It is associated with a unique predicate symbol, for instance *Automobile*(x), which holds *true* if an entity replacing the variable x is an automobile, otherwise holds *false*. Thus, an

ontological category can be described by providing necessary and sufficient properties for membership that an entity needs to satisfy. Let us denote by $\mathcal{P}$ a set of properties expressed by unary predicates, thus an ontological category U is defined by a set of properties $\Omega_U \subset \mathcal{P}$. That is, $U(x) = \bigwedge_{P \in \Omega_U} P(x)$.

3.2.2 An Inheritance Relation between Ontological Categories

An ontological relation is placed between the ontological categories to model a certain relation existing between entities that belong to the categories. Among many relations available in an ontology, an *inheritance* relation is the main focus of this work as it expresses an inheritance of properties from one ontological category to another. Different ontologies use different symbols (*e.g., IS-A*, *subClassOf*, *subCategoryOf*) to denote this inheritance relation, here we use *IS-A*, as it is the most common symbol among many ontologies. Given a set of categories denoted by *C*, the *IS-A* relation over *C* is defined by a subset of $C \times C$ such that

- $(U, V) \in$ *IS-A* $\implies \Omega_V \subset \Omega_U$,
- $(U, V) \in$ *IS-A* $\wedge (V, W) \in$ *IS-A* $\implies (U, W) \in$ *IS-A*.

For each ordered pair $(U, V) \in$ *IS-A*, *U* is a sub-category of *V* and *V* is a super-category of *U*. That is *U* is described by a set of properties inherited from *V* along with its own additional properties. Let us assume that there is a category $\top \in C$, where $\forall U \in C, (U, \top) \in$ *IS-A* as a super-category of all categories in this relation. A graphical representation of *IS-A* is defined by $G_{\textit{IS-A}} = (C, \textit{IS-A})$, which is a directed acyclic graph. A set of categories *C* is represented by a set of vertices. For each ordered pair $(U, V) \in$ *IS-A*, a directed edge is placed from a vertex associated with a sub-category *U* to a vertex associated with its super-category *V*. The $\top$ concept is a root of this graph as its associated node has no incoming edges. A transitive reduction of $G_{\textit{IS-A}}$ is assumed so that a metric to measure the similarity between categories from this structure is well defined. This reduction is denoted by $G'_{\textit{IS-A}}$, which is a directed graph with the smallest number of edges such that for every path between vertices in $G_{\textit{IS-A}}$, $G'_{\textit{IS-A}}$ also has a path between those vertices [58].

3.3 The Measurement of Similarity between Categories

In this work, the notion of similarity between two categories is relative to the number of properties shared between them, which can be estimated from the structure of $G'_{IS\text{-}A}$. Denote this similarity measurement by $s_{ont} : C \times C \rightarrow [0, 1]$. Note that this measurement provides merely a qualitative information useful for comparison not the exact number of properties shared between two categories.

3.3.1 Properties of the Similarity Measurement

The following are properties, in which s_{ont} needs to capture.

- The maximum similarity between two categories is reached when they are identical. Therefore, $\forall U, V \in C, s_{ont}(U, V) = 1 \iff U = V$.
- The similarity between categories is the same no matter the order of the comparisons, that is $\forall U, V \in C, s_{ont}(U, V) = s_{ont}(V, U)$.

There are two components involved in computing this metric: specificity of a category and the number of properties that two categories have in common. The computation of these components from $G'_{IS\text{-}A}$ is described in the following sections.

3.3.2 Estimation of the Category's Specificity

Since a sub-category contains a greater number of properties than its super-concept, we say that the former is more *"specific"* than the latter [59]. A measurement of category specificity is denoted by $\gamma : C \rightarrow [0, \infty)$ such that $(U, V) \in$ *IS-A* $\implies \gamma(U) > \gamma(V)$. That is $\top$ is the least specific category. Regarding the transitive property of *IS-A*, the specificity of categories monotonically increases over $G'_{IS\text{-}A}$ from the root ($\top$) to leaves (categories without any sub-categories).

A straightforward way to measure specificity of a category is to count the number of properties used to describe the category. This information, unfortunately, may not be available as it is difficult to enumerate all necessary and sufficient properties to describe a particular category. Two general approaches have been developed to get around this issue in estimating specificity of a category.

3.3.2.1 *Using an Extra Source of Information*

This type of approach [60, 61] relies on an extra source of information, such as a corpus of texts, where the associated categories or classes of these texts are annotated. Generally, the specificity of a category is estimated from the number of its occurrences in a given corpus where the greater the number of its occurrences, the lower the value of its specificity. These approaches, however, require some pre-processing steps especially in annotating texts with their associated categories. The accuracy of the estimated specificity depends directly on this step. Moreover, the corpus or the entities' collection used in these approaches need to be large enough and particular to a set of categories so that it provides unbiased estimation of the specification and covers all the categories.

3.3.2.2 *Using a Graphical Representation*

This type of approach avoids the difficulties of the former approaches by estimating the specificity of a category directly over the graphical representation of *IS-A* relation. The idea is that the specificity of a category can be estimated by the location of its associated node in $G'_{IS\text{-}A}$. This work follows this idea in defining γ. Consider the transitive property, a depth of a vertex in $G'_{IS\text{-}A}$ can be used to estimate the specificity of an associated category [62, 63] such that the deeper the vertex, the greater the category's specificity. We denote by δ_U the depth of a vertex associated with a category U, which is determined by

$$\delta_U = |\text{nodes on the shortest path from } U\text{'s node to the root}|. \tag{3.1}$$

The specificity of U is thus defined by

$$\gamma(U) = \begin{cases} 1 & \text{if } U = \top, \\ \delta_U & \text{otherwise.} \end{cases} \tag{3.2}$$

The main drawback of such a simple specificity estimator is that categories with identical depth will have the same amount of specificity. More refined, yet computationally expensive specificity

measurements [64, 63, 62] have been developed by taking other structural properties of $G'_{IS\text{-}A}$ into account.

3.3.3 Determining the Number of Common Properties between Categories

There are two general cases to be considered when determining the number of properties shared between any categories $U, V \in C; U \neq V$ from the *IS-A* relation.

3.3.3.1 Two Categories Are Comparable

If $(U, V) \in$ *IS-A*, they are comparable. The number of properties that *U* and *V* have in common then can be estimated by the specificity of the super-category, which is the specificity of *V* in this case.

3.3.3.2 Two Categories Are Incomparable

If $(U, V) \notin$ *IS-A*, they are incomparable. The number of properties shared between two incomparable categories can be approximated from other categories, called common ancestors, in which both categories derive the properties from. A set of *U*'s ancestors is defined by $a_U = \{W | W \in C, (U, W) \in \textit{IS-A}\}$. A set of common ancestors of *U* and *V* is thus described by $CA_{U,V} = a_U \cap a_V$. Note that every pair of incomparable categories has at least $\top$ as one of their common ancestors. Among all of their common ancestors, the number of properties shared between incomparable categories is estimated from the number of properties that describes the most specific common ancestor as it is the upper bound. If $W \in CA_{U,V}$ is the most specific common ancestor of incomparable categories *U* and *V*, the number of the properties that they have in common is defined by the specificity of *W*.

3.3.4 Defining s_{ont} over $G'_{IS\text{-}A}$

Let us denote by $MSCA_{U,V}$ the most specific common ancestor of categories U and V. Using $G'_{IS\text{-}A}$, the $MSCA_{U,V}$ can be retrieved from their associated vertices such that

$$MSCA_{U,V} = \begin{cases} U & \text{if } U = V, \\ \arg\min_{U,V}(\delta_U, \delta_V) & \text{if } U, V \text{ are comparable}, \\ \arg\min_{W \in CA_{U,V}}(\delta_{W,U} + \delta_{W,U} + \delta_W) & \text{if } U, V \text{ are incomparable}, \end{cases} \tag{3.3}$$

where $\delta_{W,U}$ is the number of nodes on the shortest path from a vertex associated with U to a vertex associated with W in $G'_{IS\text{-}A}$.

The simplest expression of $s_{ont}(U, V)$ is the ratio between the specificity of their *MSCA* and the maximum specificity of all categories (*i.e.,* the maximum depth among all vertices in $G'_{IS\text{-}A}$). However, this simple expression suffers from a restriction, in which pairs of categories that have the same *MSCA* or their *MSCA* have the same depth will yield the same value of s_{ont}. In order to alleviate this limitation, the specificities of U and V are taken into account. The definition of s_{ont}, which satisfies both conditions explained in Section 3.3.1, is

$$s_{ont}(U, V) = \frac{2 \times \delta_{MSCA_{U,V}}}{\delta_U + \delta_V}. \tag{3.4}$$

Note that this revised definition of s_{ont} coincides with the similarity measurement presented by Wu and Palmer [65], and Lin [66]. This metric estimates from $G'_{IS\text{-}A}$ the qualitative information of how many properties U and V have in common with respect to the number of properties used to describe each of them.

3.4 Determining the Similarity between Entities

In order to use the Equation (3.4) in estimating the similarity between two entities, the ontological categories that contain these entities need to be identified. Note that an entity can be a member of many categories. The membership information may be available in some ontologies, otherwise

a textual matching between a symbol of an ontological category and a symbol representing an entity may be used to identify the corresponding categories. We denote by $C_u, C_v \subset C$ the sets of categories containing the given entities *u* and *v*, one for each entity. The similarity between *u* and *v* can be determined by

$$s'_{ont}(u, v) = \max_{U \in C_u, V \in C_v} s_{ont}(U, V). \tag{3.5}$$

To reduce the number of comparisons, the most specific category may be used as a representation such that $\arg\max_{U \in C_u} \delta_U$ and $\arg\max_{V \in C_v} \delta_V$ are representations of C_u and C_v, respectively.

A conceptual distance between two entities is simply defined by reversing their corresponding similarity measure such that

$$d_{ont}(u, v) = 1 - s'_{ont}(u, v). \tag{3.6}$$

That is, the higher the similarity measure between the entities, the shorter the conceptual distance between them.

3.5 Realization of Similarity Measurement between Categories with SPARQL

Since knowledge codified in an ontology is fairly large, $G'_{IS\text{-}A}$ induced from the ontology may contain hundred of thousand of nodes and edges. Loading and processing this graph locally may be challenging, as it requires a large memory space and a high computational power. Conveniently, an ontology provides an online service, which allows us to query an ontological knowledge. One common query language available across ontologies, which use Resource Description Framework (RDF) to express knowledge, is SPARQL. Even though this query language has some limitations, it allows us to retrieve information from an ontology on the fly without any pre-processing steps or loading the whole category organization to a local machine. The examples of SPARQL queries over DBpedia [39]—an ontology counterpart of Wikipedia— are presented to compute s_{ont}, where the symbol used to denote the inheritance relation between categories is "$skos: broader$".

The computation of s_{ont} is composed of two main components: finding the *MSCA* of two given categories, and computing the depth of a given category to determine its specificity. The SPARQL query in Listing 3.1 retrieves common ancestors whose distance over "$skos: broader$"

to the given category denoted by *Coal* and the given category denoted by *Natural_gas* are 1 (the parameter {0, 1}). If there is no category that satisfy this condition, this query returns nothing. This query is iteratively submitted to DBpedia's endpoint with the incremented distance until at least one category is returned. The returned category acts as the most specific common ancestors of two given categories. The query in Listing 3.2 asks DBpedia whether the distance from the given category denoted by *Natural_gas* to the category denoted by *Contents*, which is given as a dummy $\top$ [67], over "$skos{:}\ broader$" is 1 (the parameter {0, 1}) or not. This query is iteratively submitted to DBpedia's endpoint with the incremented distance until *true* is returned, indicating that the number that is submitted with the recent query is the depth of the category. If the query in Listing 3.1 returns more than one categories, the query in Listing 3.2 may be used to break a tie such that the category with the maximum depth is returned as the *MSCA*.

Listing 3.1: Retrieving the most specific common ancestors of two categories

```
prefix category: <http://dbpedia.org/resource/Category:>

select distinct ?super where {
?super (^skos:broader){0,1} category:Coal, category:Natural_gas
}
```

Listing 3.2: Computing depth of a category

```
prefix category: <http://dbpedia.org/resource/Category:>
ask where{
category:Contents (^skos:broader){0,1} category:Natural_gas
}
```

Applying these queries to another ontology can be done by changing the symbol denoting the inheritance relation, and the symbol denoting $\top$. For instance, in YAGO [40] the symbol of the inheritance relation is $\mathrm{rdfs{:}\,subClassof}$. This relation connects Yago's classes whose the prefix are *<http://dbpedia.org/class/yago/>*. The $\top$ of this relation is *rdfs:Resource*. The algorithm to compute s_{ont} using these queries is shown in Algorihtm 1

Algorithm 1: Computing s_{ont} between ontological categories with SPARQL

input: U, V = the symbols denoting two given ontological categories
O = SPARQL endpoint of an ontology
R = a symbol denoting the inheritance relation
T = a symbol denoting $\top$

Output: s_{ont} between U and V

$\delta_U, \delta_V \leftarrow 0$
$d_U, d_V \leftarrow$ False
while *not* d_U **do**
    /* Computing depth of U using query in Listing 3.2 */
    $\delta_U \leftarrow \delta_U + 1$
    $d_U \leftarrow$ **SPARQLDepth**(U, R, T, δ_U, O)
while *not* d_V **do**
    /* Computing depth of V using query in Listing 3.2 */
    $\delta_V \leftarrow \delta_V + 1$
    $d_V \leftarrow$ **SPARQLDepth**(V, R, T, δ_V, O)
/* Retrieving common ancestors of U and V using query in Listing 3.1 */
$MSCA_{U,V} \leftarrow \emptyset$
$d_{ca} \leftarrow 0$
while $MSCA_{U,V} = \emptyset$ **do**
    $d_{ca} \leftarrow d_{ca} + 1$
    $MSCA_{U,V} \leftarrow$ **SPARQLCA(**U**,** V**,** R**,** d_{ca}**,** O**)**
/* Computing depth of $MSCA_{U,V}$ */
$\delta_{MSCA} \leftarrow 0$
foreach $W \in MSCA_{U,V}$ **do**
    $\delta_W \leftarrow 0$
    $d_W \leftarrow$ False
    while *not* d_W **do**
        $\delta_W \leftarrow \delta_W + 1$
        $d_W \leftarrow$ **SPARQLDepth**(W, R, T, δ_W, O)
    if $\delta_W > \delta_{MSCA}$ **then** $\delta_{MSCA} \leftarrow \delta_W$
$s_{ont} \leftarrow (2 * \delta_{MSCA})/(\delta_U + \delta_V)$
Return s_{ont}

3.6 Summary

This chapter describes the notion of relevance between real world entities at the conceptual level. This relevance information is quantified by the conceptual distance obtained from a category organization in an ontology. The reusable SPARQL queries for computing this metric on different ontologies are presented. Exploiting this metric in addressing a classical problem in machine learning—feature selection—will be explained in the next chapter, as this problem centers around the notion of relevance or distance for evaluating individual features or a feature subset.

4. FEATURE SELECTION VIA ANALYSIS OF RELEVANCE AND REDUNDANCY

This chapter describes how to employ the conceptual distance computed from an ontology to address a classical problem in supervised learning—feature selection. Feature selection becomes a necessary component in supervised learning to help improve the generalization of a model being built, especially in a learning problem to which a few number of examples are available or a large number of input features is used to describe an example. The idea of this technique centers around choosing relevant and removing redundant input features. Traditionally, data of features are used to evaluate their relevance and redundancy. This work introduces the criteria defined over the semantic information obtained from an ontological structure in assessing relevant and redundancy of input features. The details of a feature selection framework using these criteria along with its implementation with the large-scaled ontologies are explained. Small-scaled experiments were conducted to show the usefulness and effectiveness of the derived ontological information in feature selection.

4.1 Feature Selection

Before describing the new criteria for evaluating input features and feature selection framework developed in this work, some background and ideas underlying feature selection are worth to discuss here. Given a set of *N* training examples generated from an unknown function *f*, each example is represented by a set of *d* input features $\mathbf{X} = \{X_1, ..., X_d\}$ and an output feature denoted by Y. Feature selection concerns the problem of choosing the "best" subset of *p* features out of the *d* dimensions, then pruning the rest. There are two main ingredients derived from feature selection that play an important role in improving the generalization of a model being built. First, sparsity is imposed as a hypothesis for building a model contains only *p* number of input features. This factor simplifies a hypothesis to a certain degree, which may prevent unnecessary or noisy input features to be included in a model. Second, the chosen feature subset helps pinpoint a particular hypothesis, which believes to be very similar to the true function *f*. While it is straightforward to

specify the number of selected features, evaluating the goodness of individual features or a feature subset is the real challenge in a feature selection problem.

4.1.1 Metrics Used to Evaluate Features

Feature selection algorithms can be distinguished into three different categories with respect to the metrics used to evaluate the features.

4.1.1.1 Wrappers

The first category, wrapper methods use the prediction accuracy of a predetermined learning algorithm to evaluate the goodness of a feature subset. These methods are computationally expensive as a new model needs to be trained and evaluated for each subset. They also require a sufficient amount of data as the whole input examples have to be split into a training set for training a model and validation set for evaluation. However, wrappers usually provide the best feature subset for a particular learning algorithm. Stepwise regression [19] is an example of the wrapper method that has been used in many disciplines.

4.1.1.2 Filters

The second category, filter methods rely on the proxy measurements that capture certain properties of features useful for assessing their merits rather than using the prediction accuracy, . The useful properties for evaluating the goodness of a feature, such as distance, information, dependency, and consistency [68], are measured directly from the feature's data. These methods are typically performed as a preprocessing step separated from a learning algorithm. Therefore, they provides a generic selection of features that is not tuned for or by a specific learning algorithm. Even though filters yield sub-optimal feature subset for a given learning algorithm, they are computationally less expensive than the wrappers. FOCUS [69] and Relief [70] algorithms are the classical examples of the filter approach. Filters methods have also been used as a preprocessing step for wrapper methods in order to enable the latter to work with a large number of features.

4.1.1.3 Embedded

The third category, embedded methods incorporate feature selection as part of the training process. Decision tree algorithms [71, 72] are examples of these methods. At each recursive step, a new feature is selected with respect to a certain criterion *e.g.,* information gain), and added to the current tree. Regularization techniques, such as Lasso [73] and Ridge Regression [74], are another important embedded feature selection. These techniques aim to fine-tune model complexity by augmenting a prediction accuracy of a predetermined learning algorithm with an additional penalty constraining the algorithm toward low complexity models. Even though the embedded methods are similar to the wrappers as they are usually specific to particular learning algorithms, they make a better use of the available data by not needing to split the input examples into the training and the validation sets. These approaches also reach the solution faster than wrappers by avoiding retraining a model from scratch for every feature subset investigated.

4.1.2 How to Evaluate a Feature Set

Feature selection methods can also be divided into two classes based on how they evaluate a feature subset.

4.1.2.1 Individual Evaluation

The individual evaluation also known as feature weighting or feature ranking [15], assessing individual input features and assigning them scores or weights according to their degrees of "relevance" to the output feature *Y*. The definitions of "relevance" and choices of its measurement vary from work to work. For example, Guyon and Elisseeff [16] uses the measurement of mutual information (*M*I) and defines that X_i is *"individually irrelevant"* if and only if for some threshold $\epsilon > 0,\ \mathrm{MI}(\mathbf{x}_i, \mathbf{y}) \leq \epsilon$, where $\mathbf{x}_i, \mathbf{y}$ are the vector of data of an input feature X_i and the output feature Y. Whereas, Yu and Liu [75] use the symmetrical uncertainty (*S*U) and determine that X_i is relevant if $\mathrm{SU}(\mathbf{x}_i, \mathbf{y}) > \epsilon$. To choose a feature subset, the individual features are ranked with respect to their relevance, in which a suitable cut-off point in the ranking for selecting a certain number of features is specified either by a user or via the cross-validation technique.

Both theoretical analysis and empirical evidence show that along with irrelevant features, redundant features also affect the speed and accuracy of learning algorithms and thus should be eliminated as well [17, 14, 76]. Evaluating each feature individually, however, is incapable of removing redundant features because they are likely to have similar scores. That is, as long as features are deemed relevant, they will all be selected even though many of them are redundant to each other.

4.1.2.2 Subset Evaluation

Apart from redundant features, examples presented by Guyon and Elisseeff [16] also demonstrate that a feature that is completely useless by itself can provide a significant performance improvement when taken with others. That is, the p best features are not necessary to be the best p features, which points out that individual evaluation is usually sub-optimal. In order to address these problems, the second class, subset evaluation [17, 14, 77, 78, 79, 80] aims to assess a feature subset as a whole. In general, this approach iteratively evaluates a candidate subset of features, which is generated based on a certain search strategy. An evaluation metric that takes into account both redundancy and relevance of all features in the current candidate subset is used and then compared with the previous best subset with respect to the metric. If a new subset turns out to be better, it replaces the previous best one. The process of subset generation and evaluation is repeated until a given stopping criterion is satisfied. In order to avoid an exhaustive search over all possible feature subsets, various heuristic strategies (*e.g.,* greedy sequential search, best-first search, genetic algorithm) along with stopping criteria (*e.g.,* maximum number of features in the subset, maximum allowed running time) are exploited. Even though subset evaluation is able to find an optimal feature subset, it is not scaling well to a dataset containing a large number of input features [75].

4.2 New Feature Selection Method

The feature selection method introduced in this dissertation is a filter approach, where each input feature is examined individually with the relevance criterion defined by the metric computed

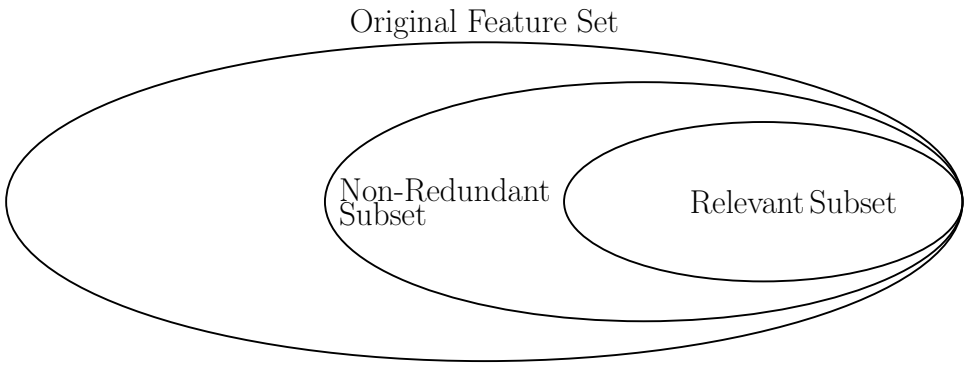

Figure 4.1: Overview of the feature selection method introduced in this work

from an ontology. To overcome the shortcoming of a traditional filter approach, which is incapable of handling redundant features, a feature clustering is explicitly added into the framework to remove redundant input features. A criterion used for clustering input feature is also defined by the conceptual distance derived from an ontology. The overview of how this method sequentially shrunk the original input feature set to obtain the relevant feature subset is shown in Figure 4.1. Two main components of the framework—relevance analysis and redundancy analysis—are described separately in the following sections as they can work independently from one another. A combination of these two components yields a filter feature selection framework that can handle both feature relevance and redundancy.

4.3 Relevance Analysis

Choosing the input features that are relevant to an output feature has been the main focus of a feature selection problem [17, 14, 75, 76]. The analysis of feature relevance helps in improving the generalization of a model being built in two complementary ways. First, by selecting only the relevant input features, the irrelevant ones are discarded from being used to define a hypothesis, thereby the sparsity is imposed. Second, as merely relevant features remain, a hypothesis pinpointed by these features is supposed to be similar to the true function that generates the data. The main component of relevance analysis is a criterion for assessing the relevance of an input feature. This work introduces a new relevance criterion defined by the measure obtained from an ontology.

4.3.1 Relevance Criterion

Definitions and criteria for evaluating relevance of an input feature have been introduced in many works [75, 77, 78]. They are typically expressed in terms of a relationship at the data level between an input feature and an output feature. This relationship is quantified by a certain measurement such as mutual information, correlation coefficient. By assuming that a feature is a realization of a particular entity, this work aims to lift the decision of feature relevance at the conceptual level through the relevance of their associated entities. As pointed out in the former chapter that the relevance of entities is defined in terms of the similarity in properties between the entities and this information can be obtained from an ontology via the measurement (3.6). Using this metric, a criterion to heuristically determine relevance of an input feature at the concept level is defined such that

Definition 4.3.1. Given two entities σ_i and σ_Y corresponding to an input feature $X_i \in \mathbf{X}$ and an output feature *Y*, respectively, X_i is *relevant* to *Y* if $d_{ont}(\sigma_i, \sigma_Y) < \epsilon$, where $\epsilon \in [0, 1]$ is a given threshold.

Given a set of entities Σ associating with a set of input features $\mathbf{X}$, where $\sigma_i \in \Sigma$ is a corresponding entity of an input feature $X_i \in \mathbf{X}$, a relevant feature subset $\mathbf{X}_R \subset \mathbf{X}$ is thus defined by

$$\mathbf{X}_R = \{X_i | X_i \in \mathbf{X},\, \sigma_i \in \Sigma, d_{ont}(\sigma_i,\, \sigma_Y) < \epsilon\}. \tag{4.1}$$

More precisely, this criterion tends to select an input feature whose corresponding entity shares many properties with an entity associated with the output feature *Y*. Noting that this criterion is defined under the assumption that there is no input feature that is associated with the same entity as *Y*, otherwise such an input feature is deemed relevant with respect to the criterion as its d_{ont} becomes zero. However, this redundant feature does not provide any useful information in predicting the value of *Y*.

4.3.2 Effect of Parameter ϵ

The main component of the above criterion is the parameter ϵ, which specifies the relevance threshold for selecting the input features. This parameter also contributes to the degree of sparsity such that setting a high value to ϵ (≈ 1) would yield a large $\mathbf{X}_R$ as more number of features can satisfy the specified criterion. Applying such an unrestricted criterion does not increase the sparsity of a model being built, and may include irrelevant features into $\mathbf{X}_R$. On the other hand, if a low value is assigned to ϵ (≈ 0), the specified criterion becomes more restricted leading to a small $\mathbf{X}_R$ as only a few features can pass through. Even though it helps improve the sparsity of a model, some relevant features may be left out.

To be able to separately study the effect of sparsity and the goodness of the chosen features in $\mathbf{X}_R$, the parameter ϵ is replaced by a new parameter $p \in (0, |\mathbf{X}|)$, which specifies the number of input features to be selected (*i.e.,* the degree of sparsity). Given *p*, constructing $\mathbf{X}_R$ is done by sorting the individual input features in ascending order with respect to their corresponding d_{ont}, the top-*p* features are then selected to form $\mathbf{X}_R$. Both sorting and parameter *p* work together in the same manner as ϵ to manage both sparsity and relevance levels of a selected feature subset.

4.3.3 Computing Conceptual Distance between Features

In order to measure the relevance of input features at the conceptual level using the measure (3.6), each feature (including the output one) must be mapped to its corresponding ontological categories. This work assumes that a feature is associated with a label providing a short description in natural language about what the feature really measures. Given such a label denoted by ℓ, the simplest way to map a feature to a set of ontological categories, which contains an entity associated with the feature, is done through a textual matching. That is among all ontological categories expressed in an ontology, ones whose symbols contain words that appear in the feature's label are retrieved. Searching for a set of textual-match ontological categories is done by tokenizing ℓ to a set of words, in which stopwords (*e.g.,* the, for, *etc.*), numbers, special characters, and punctuation are removed. For each remaining word *w*, the ontological categories whose symbols contain *w* is

retrieved via a particular textual matching operation such as regular expression. A set of ontological categories denoted by C_ℓ, which is correspond to the label ℓ is defined by $C_\ell = \bigcup_{w \in W} C_w$, where W is a set of the remaining words and C_w is a set of ontological concepts whose symbol contains a word w. If $C_\ell = \emptyset$, W my be augmented by including the synonyms, hyponyms, and hypernyms of each word $w \in W$, and then running the mapping again.

Some ontologies provide a service to search for the ontological categories that match an input query. For example, DBpedia has a look up service[1] to obtain DBpedia categories corresponding to a querying word or phrase. Some software libraries are also developed to serve this purpose. Sematch [81], for instance, provides a function that can map from a word or phrase to the associated Yago's classes. Thus, the textual matching operation may be implemented by exploiting such an online service or a certain function from an existing library. The workflow of computing d_{ont} from given labels of two features is shown in Figure 4.2.

4.3.4 Selecting a Relevance Feature Subset

The algorithm of feature selection that assesses relevance of features with respect to relevance of their corresponding entities via the metric d_{ont} is described in Algorithm 2.

There are two points worth noting from this algorithm. First, the choices for implementing the **FindConcepts** function depend on a given ontology as different ontologies provide different ways to search for ontological categories or classes that match an input query. Second, the computation of the nested *for* loop in lines 4 to 12 can be reduced by using the most specific category as a representation for a set of categories C_ℓ and a set of categories C_Y instead.

4.3.5 Experimental Results

We argue about the idea of using the metric computed from an ontology in selecting relevant input features through three separate experiments, in which the usefulness and effectiveness of this measure in improving the prediction performance of a model being built is the matter of interest. The first experiment aims to show how well the metric derived from an ontology does in feature

[1] `http://lookup.dbpedia.org/api/search.asmx/KeywordSearch`

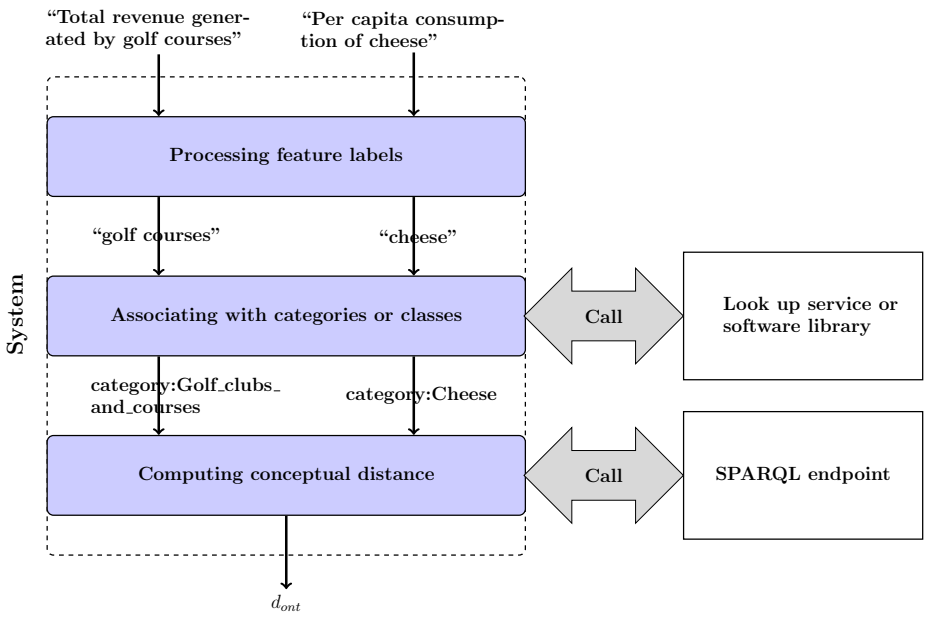

Figure 4.2: The workflow of computing $distance$ between two features using their labels. There are three components working together sequentially from top to bottom. Given labels of features, the first component remove stopwords, numbers, punctuations to clean up the labels. The remaining texts from this component are sent to the second component to find corresponding categories from a given ontology through its look up service. Then the third component use SPARQL queries to compute the conceptual distance using categories output from the second component.

Algorithm 2: Selecting p relevant features

input: $\mathcal{L}$ = a set of input features' labels
ℓ_Y = a label of the output feature Y
p = number of selected features
O = SPARQL endpoint of an ontology
R = a symbol denoting the inheritance relation
T = a symbol denoting $\top$
$\mathbf{X}$ = a set of input features

Output: $\mathbf{X}_R$ = a set of p relevant features

/* This algorithm use Algorithm 1 to compute ϱ denoted by **SPARQL**$_\varrho$ */

$\mathbf{X}_r \leftarrow \emptyset$
$S \leftarrow []$
$C_Y \leftarrow$ **FindConcepts**(ℓ_Y, O)
foreach $\ell \in \mathcal{L}$ **do**
    $C_\ell \leftarrow$ **FindConcepts**(ℓ, O)
    $Q \leftarrow []$
    foreach $U \in C_\ell$ **do**
        foreach $V \in C_Y$ **do**
            $s_{ont} \leftarrow$ **SPARQL**$_{s_{ont}}(U, V, R, T)$
            add s_{ont} into Q
    $s'_{ont} \leftarrow$ **max**(Q)
    add $(1 - s'_{ont})$ into S
Xsorted $\leftarrow$ **SORT**(**X**, S)
$i \leftarrow 0$
while $i < p$ **do**
    add Xsorted[i] into $\mathbf{X}_R$
    $i \leftarrow i + 1$
Return $\mathbf{X}_R$

selection compared to the traditional source of information for feature selection—data. The second experiment was conducted to demonstrate the informativeness of the ontological measurement in selecting relevant features. The third one addresses the question concerning whether our approach can be used across different ontologies.

4.3.5.1 Datasets for Experiments

Two datasets were constructed to conduct experiments in this section. These datasets will also be used to perform experiments in other sections throughout this chapter. The first dataset is motivated by the research of Nelson and Sprecher [1], which builds a model of nuclear reliance in a given country to help understand civil nuclear proliferation. The 60 attributes of the 83 countries (out of 86 countries described in the research) are used as the input features to predict the share of total electricity used that is generated by nuclear power, which is a realization of the entity *nuclear reliance*. Data and labels of these features were collected from the CIA Factbook.[2].

The second dataset is inspired by recent national concern over violent crimes and gun control debates. Follow the research by Monuteaux et. al. [82], which studies an association between firearm ownership and violent crime in the U.S., we aim to build a model to predict the number of violent crimes per states (*i.e.,* a realization of the entity *violent crime*) in the U.S. The 60 attributes of 51 states and territories (Alaska was excluded) are used as the input features. The data and labels of these features were collected from StateMaster.[3] The missing values of a feature in both datasets were handled by replacing them with the average value of the feature. Data of all input features in both datasets were standardized to the range of 0 to 1. For brevity we will refer to the first dataset using the words *Nuclear Reliance* and the second one using the words *Violent Crime*.

4.3.5.2 Effectiveness and Usefulness of Ontological Relevance

This experiment illustrates the effectiveness of entity relevance obtained from an ontology in choosing relevant input features for building a model from each dataset. The main question we want to answer by this experiment is that given the same number of selected features (parameter p),

[2] `https://www.cia.gov/library/publications/download/`
[3] `http://www.statemaster.com/index.php`

how well the information derived from an ontology does in choosing relevant features compared to the data? If the prediction accuracy of a model built from a feature subset selected by an ontology is not so different from one built by the data, this indicates the usefulness of the metric derived from an ontology in addressing the problem of feature selection.

For each dataset, this experiment was conducted by splitting the data examples into training and test sets, in which the test set contains 30% of the examples. A linear model was learned from the training set using a standard linear regression method and its prediction accuracy was quantified on the test set via the residual sum of squares (RSS). The parameter p was incremented from 3 to 60 to specify the number of input features to be selected (*i.e.,* sparsity level). For each specific p, the process of learning a model was repeated 50 times, the mean and standard deviation of RSS from 50 iterations were collected for the plots. The baseline method for comparison was developed by sorting input features in descending order with respect to the mutual information between their data and the data of the output feature and then selecting the top-p features. For this first experiment, DBpedia [39] ontology was used. The experimental results obtained from both datasets are shown in Figure 4.3 and Figure 4.4, for each dataset. Note that only the results obtained when set p in the range from 3 to 30 are shown to emphasize the performance at low p.

From both figures, we can see that as the parameter p increases, the prediction accuracy of a model decreases (RSS increases) as it becomes complex, and more number of irrelevant features may be added into a model. Despite using only features' labels and a certain ontological structure, the prediction accuracy of a model built from this approach is not so different from one produced by the data. These results point out the usefulness of the metric obtained from DBpedia for feature selection. Since this behavior holds on both datasets, it is the evidence showing that this metric can be applied across problem domains. At certain values of p, we can also see that this approach produces better models than using the traditional data-driven measurement in selecting relevant features, which indicates the effectiveness of this approach.

The common trend in Figure 4.3 and Figure 4.4 points out that both sparsity and relevance information obtained from an ontology are complementary. If p is low, the prediction accuracy

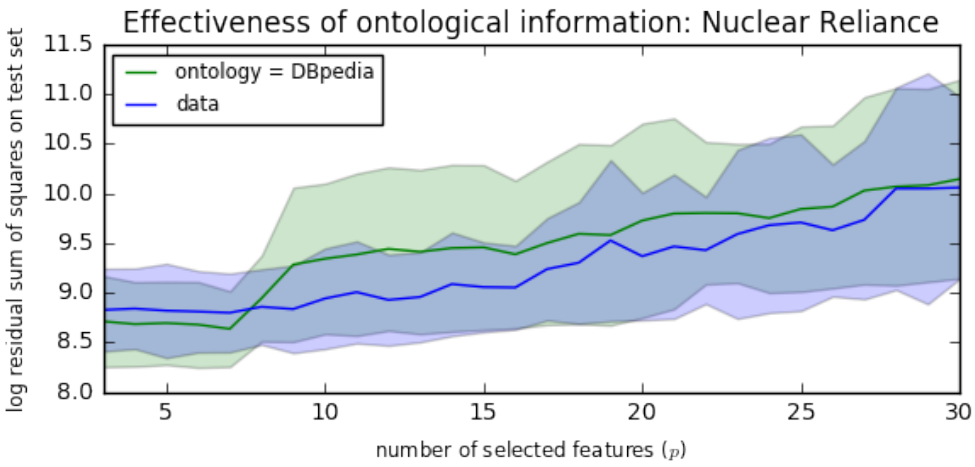

Figure 4.3: This plot compares the prediction accuracy on test sets of models built from feature subsets selected by our approach (green line) and subsets selected by the the data (blue line) on different levels of sparsity for the *Nuclear Reliance* dataset. DBpedia is used as a source to compute feature relevance.

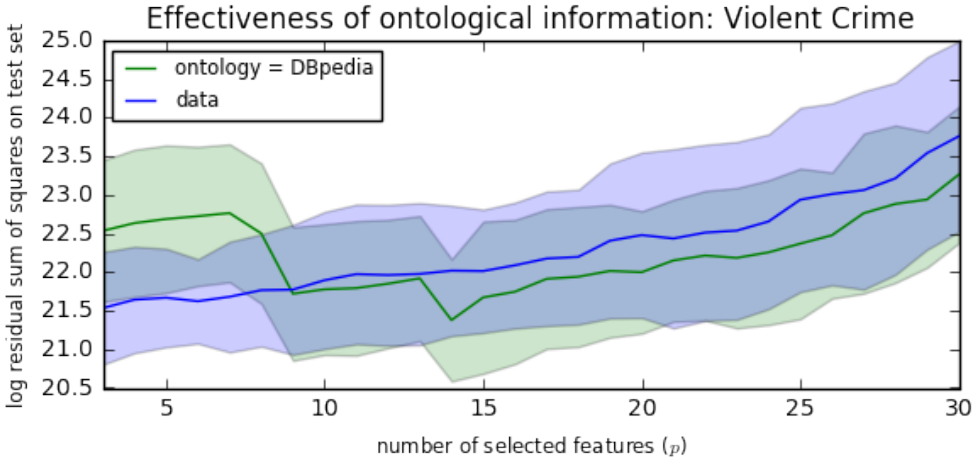

Figure 4.4: This plot compares the prediction accuracy on test sets of models built from feature subsets selected by our approach (green line) and subsets selected by the the data (blue line) on different levels of sparsity for the *Violent Crime* dataset. DBpedia is used as a source to compute feature relevance.

tends to be low as some relevant are left out to trade with the sparsity, especially in the *Violent Crime* dataset. That is the input features in the sparse subset are not enough to explain the variance of the output feature. When p is increased to a certain point, the model obtained from such a subset is the most accurate indicating that relevant features in this particular subset suffice. Increasing p from this point onward not only decreases the sparsity but also increases the number of irrelevant features, thereby the prediction accuracy of the models being built from these subsets pointing to their poor generalization. However, a critical question still remains as we do not know whether the effectiveness of this approach comes from the sparsity of the subset alone or feature relevance obtained from an ontology also plays a role. We will answer this question with the experiments in the following section.

4.3.5.3 Informativeness of Ontological Relevance

The question we want to answer in this experiment is whether the relevance information obtained from an ontology plays any role in improving prediction accuracy of a model being built. This experiment aims to show that the particular choice of features not merely the sparsity of a selected feature subset also impacts the model's generalization. More precisely, we want to know whether the relevance measure derived from an ontology really provides information that reflects the quality of an input feature. The parameter p will be hold constant in this experiment so that the only factor that matter to the prediction accuracy is which features are selected to build a model. For each dataset, the parameter p was set to the value that seems to produce the best model (*i.e.,* 7 for *Nuclear Reliance* and 14 for *Violent Crime*). To construct an alternative feature subset, the p number of features was selected randomly and uniformly without replacement from the original input feature set to construct a random feature subset, which indicates that no information has been used to select these features. A linear model was learned and tested in the same fashion as the former experiment, then the mean RSS on test sets was collected to quantify the prediction accuracy obtained from the subset. The samples of 500 random feature subsets were generated to compare with two subsets chosen by DBpedia and the data from the former experiments. The quality of a feature subset is determined by the conceptual distance of all features within the subset, which can

be computed with $\sum_{X_i \in S} d_{ont}(\ell_i, \ell_Y)$, where S is a feature subset and ℓ_i is a label of input feature X_i. The plots illustrating a relationship between the prediction accuracy of a model built from a feature subset and the total conceptual distance of features in the subset for both datasets are shown in Figure 4.5 and Figure 4.6, one for each dataset. We expect two things to happen here. First, we expect to see some form of linear pattern between the quality of a subset and the performance of a model built from the subset. That is when the total distance of features in a subset increases, the prediction accuracy of a model built from the subset should decrease (RSS increases). Second, the subset selected by DBpedia should produce very good if not the best model.

As expected, we can see some form of linear patterns from both plots, indicating the informativeness of the distance metric derived from the ontology in selecting a relevant feature subset. As irrelevant features appear in a subset, they not only increase the total distance of the whole subset but also make the prediction accuracy of a model built from the subset decrease. When comparing the quality of models produced from the random subsets with a subset selected by DBpedia (dark green circle), we can see that the subset obtained from DBpedia produces a very good or the best model (in *Nuclear Reliance*) as expected. We can also see that feature subset selected by data (blue triangle) may include irrelevant features, leading to poor performance. If the relevance measure obtained from the ontology does not provide any information about the feature relevance, this pattern would not be exhibited.

4.3.5.4 Does Our Approach Work across Ontologies?

The experiments in this section aim to address the question concerning whether the ontological background knowledge, as used in our approach, is effective across ontologies. Despite the various organizations of the ontological categories in different ontologies, we want to demonstrate that the general information used to compute the conceptual distance for relevance analysis can be obtained effectively from other ontologies besides DBpedia. After all, DBpedia contains a large and general category structure that covers numerous problem domains. The same experiments as in Section 4.3.5.2 were conducted except that Yago [40], which is another large scale ontology constructed by combining information from Wikipedia and WordNet [41], was used to compute

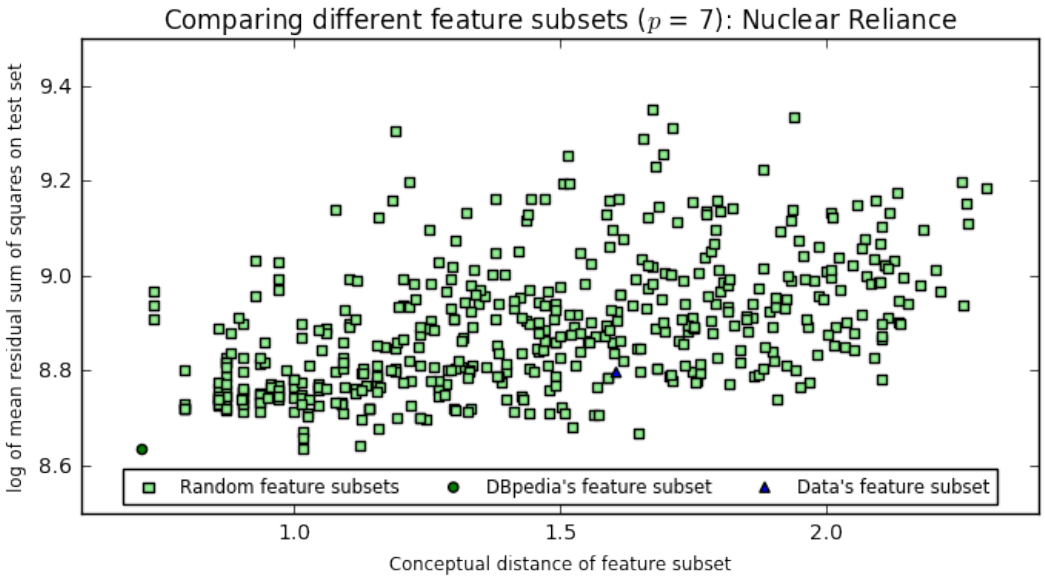

Figure 4.5: *Nuclear Reliance*, this figure shows a relation between prediction quality of a model being built and the relevance of a feature subset. A pattern can be seen as the relevance of features in a subset decreases (high distance), the prediction quality of a model learned from the subset tends to decrease (RSS increase).

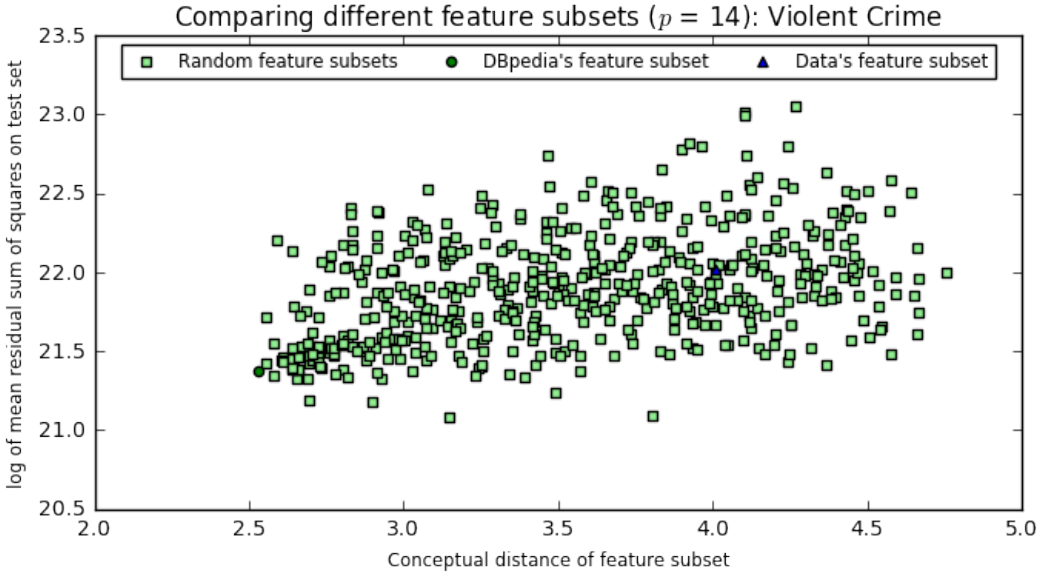

Figure 4.6: *Violent Crime*, this figure shows a relation between prediction quality of a model being built and the relevance of a feature subset. A pattern can be seen as the relevance of features in a subset decreases (high distance), the prediction quality of a model learned from the subset tends to decrease (RSS increase).

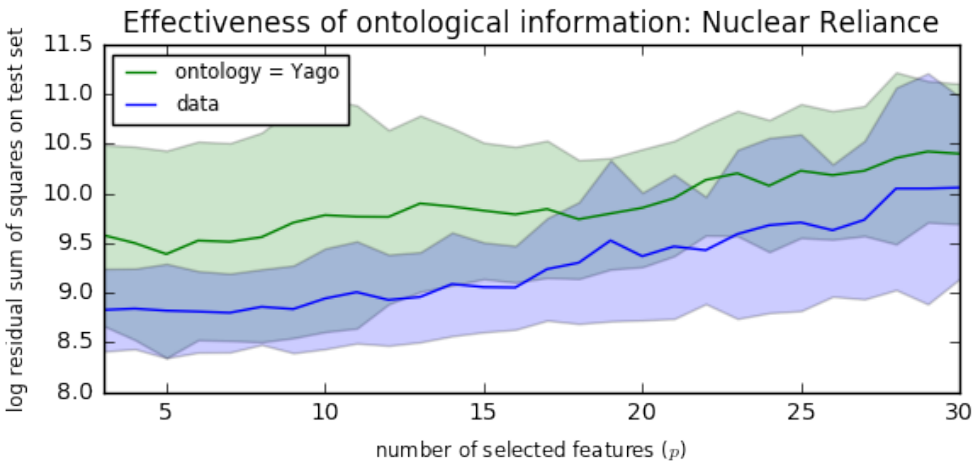

Figure 4.7: *Nuclear Reliance*, this plot compares the prediction accuracy on test sets of models built from feature subsets selected by our approach (green line) and subsets selected by the the data (blue line) on different levels of sparsity. Yago is used as a source to compute feature relevance.

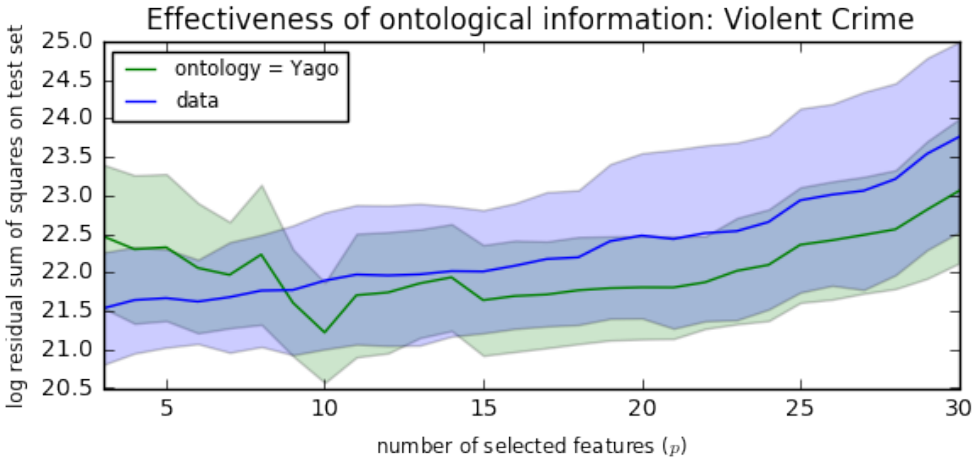

Figure 4.8: *Violent Crime*, this plot compares the prediction accuracy on test sets of models built from feature subsets selected by our approach (green line) and subsets selected by the the data (blue line) on different levels of sparsity. Yago is used as a source to compute feature relevance.

feature relevance. If the relevant information can be obtained effectively from Yago, we should see the behavior of our feature selection method similar to the plots in Figure 4.3 and Figure 4.4 for each dataset.

Comparing the plots in Figure 4.7 and Figure 4.8 to the plots in Figure 4.3 and Figure 4.4,

we can see that the behavior of our approach when using Yago does not change markedly from using DBpedia on both datasets. These results support the belief that our approach work across ontologies, but its efficiency over a particular domain may depend on the adequacy of the high-level knowledge codified in the ontology. As the results from the *Nuclear Reliance* dataset show that using Yago is not as effective as using the data or DBpedia, which points out that Yago may not provide a sufficient structure of ontological categories for this domain. Surprisingly, however both Yago and DBpedia produce very similar pattern on the *Violent Crime* dataset indicating the sufficiency of knowledge over this domain on both ontologies.

4.3.5.5 Lessons Learned from the Experimental Results

The results from Section 4.3.5.2 and Section 4.3.5.4 show that for both ontologies our approach works well on the *Violent Crime* dataset, but it is not so effective on the *Nuclear Reliance* dataset, especially with Yago. This situation points to the important question concerning whether there is a way to determine the success of our approach on a particular dataset beforehand. One obvious hint for answering this question is by studying the variation of the distance metrics computed from an ontology. The idea is that if the range of the distance measure for all input features is narrow or many input features obtain the same distance, it could be difficult to clearly distinguish relevant features from the irrelevant ones. The frequency histograms of the conceptual distances computed from DBpedia and Yago are shown in Figure 4.9 and Figure 4.10, giving the dynamic range for each dataset.

We can see from both figures that the range of the distances in the *Nuclear Reliance* dataset is narrow (0.0–0.5) compared to the range of the distances in *Violent Crime* dataset (0.0–1.0) for both ontologies. Especially in the *Violent Crime* dataset, the frequencies of the short distance is low compared to the frequencies of the long distances. Both range and frequency of the distances play an important role in successfully distinguishing the relevant features from the irrelevant ones, which lead to a better selected feature subset. Lacking these two factors, the relevance information obtained from the ontology may not help much, as shown in the *Nuclear Reliance* dataset with Yago (noting that a set of 7 shortest input features from DBpedia yields us the best model). Even

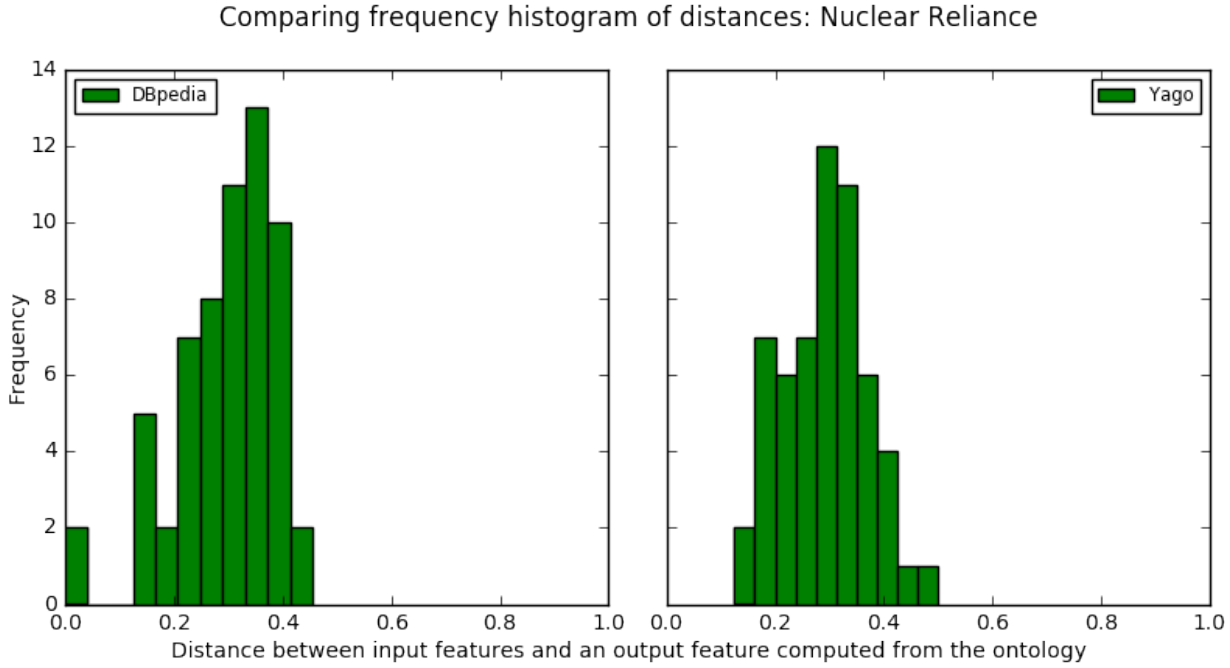

Figure 4.9: This figure compares the frequency histograms of the conceptual distance computed from DBpedia (on the left) and Yago (on the right) for all input features in the *Nuclear Reliance* dataset.

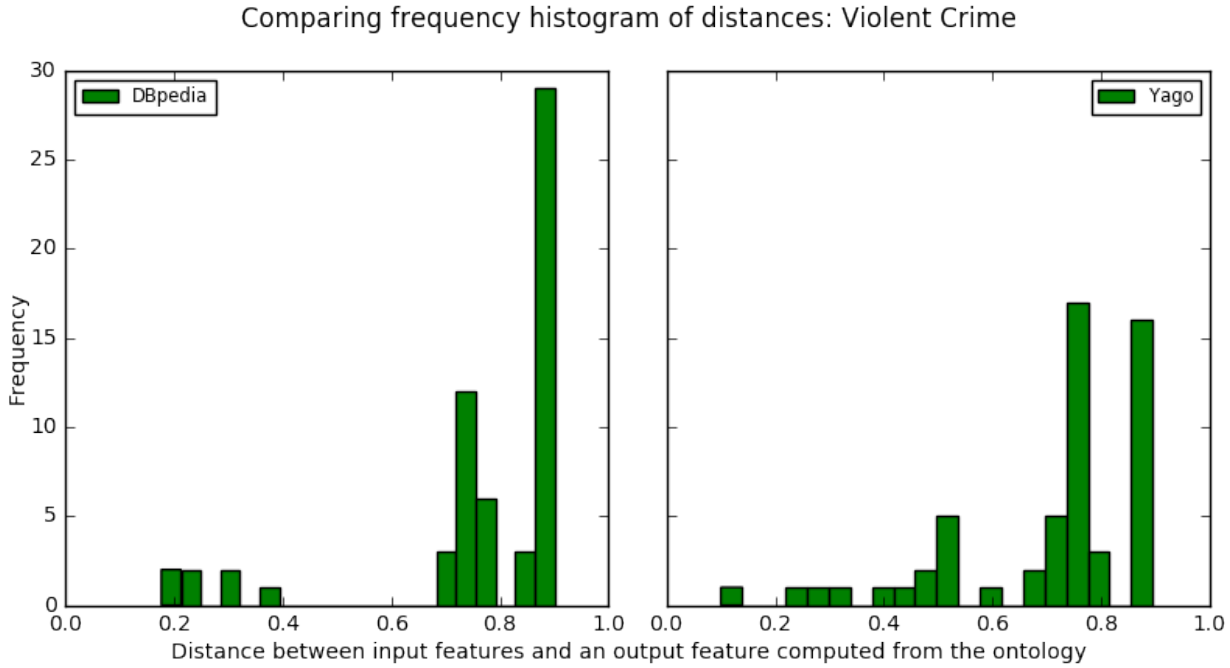

Figure 4.10: This figure compares the frequency histograms of the conceptual distance computed from DBpedia (on the left) and Yago (on the right) for all input features in the *Violent Crime* dataset.

though we do not know exactly which structural properties of an ontology account for the variation of the relevance score, we can plot such a histogram to examine in advance whether a given ontology would work with a given input dataset.

4.4 Redundancy Analysis

A large dataset usually contains many redundant input features, which typically do not provide any new information that contributes to improve the model's prediction performance. Using all redundant features imposes unnecessary complexity to a model being built, which may lead to a poor generalization. Choosing only one input feature among redundant ones should be sufficient. The notion of redundancy between two features is traditionally described in term of feature correlation. It is widely accepted that two features are redundant to each other if their values are completely correlated [75]. In practice, however, it is difficult to determine feature redundancy when a feature is correlated partially with another feature. Choosing a representation among redundant input features poses another challenge, as a certain criterion is needed in order to identify which feature is the most *suitable* one.

Here, redundancy between two input features is assessed at a conceptual level rather than at the traditional data level. The intuition is that two features are redundant if their associated entities have many properties in common. Using the metric (3.6), a criterion to heuristically determine redundancy between two input features is defined such that

Definition 4.4.1. Given two entities σ_i and σ_j corresponding to input features $X_i, X_j \in \mathbf{X}$, respectively, X_i is *redundant* to X_j if $d_{ont}(\sigma_i, \sigma_j) < \varepsilon$, where $\varepsilon \in [0, 1]$ is a given threshold.

4.4.1 Effect of Parameter ε

Similar to the definition of the relevance (see Definition 4.3.1), the main component of the redundancy criterion is parameter ε. If the value of ε is set too high (≈ 1), all input features may be determined as redundant to one another. Even though this restricted criterion enforces more sparsity, as many features will be thrown away, a lot of useful information would have lost if these features are not actually redundant. On the other hand, if a given ε is too low (≈ 0), redundant

input features may not be eliminated, trading off with minimum information loss.

4.4.2 Selecting a Non-Redundant Feature Set Using Feature Clustering

Instead of directly using the criterion described in Definition 4.4.1 to evaluate the redundancy for each pair of input features in the given feature set $\mathbf{X}$, a feature clustering technique is introduced to construct a non-redundant feature subset from $\mathbf{X}$. Given the parameter *k* specifying the number of clusters to be formed, input features are clustered with respect to the conceptual distance d_{ont}. That is the input features are clustered based on the properties that their associated entities have in common. A set of features in the same cluster are considered to be redundant to each other, whereas the sets features in two different clusters convey distinct information. One feature from each cluster is then selected as a representation of the cluster, yielding a subset of *k* non-redundant features, which is denoted by $\mathbf{X}_k \subset \mathbf{X}$. The role of the parameter *k* is similar to the parameter ε, which specifies both the sparsity and the amount of information carried by $\mathbf{X}_k$. If *k* is set too low, the consequence is similar to setting ε too high. Conversely, high *k* corresponds to low ε. The algorithm of the feature clustering is described in Algorithm 3.

Several points in this algorithm need to be elaborated further. The complexity in constructing the $|\mathbf{X}| \times |\mathbf{X}|$ matrix $\mathbf{M}$ (lines 3-15), in which the component $\mathbf{X}_{i,j}$ contains the conceptual distance between features $X_i, X_j \in \mathbf{X}$, can be reduced in three ways. First, $\mathbf{X}_{i,j}$ can be set to zero by default whenever $i = j$. Second, if X_i and X_j have at least one common ontological categories, $\mathbf{X}_{i,j}$ is set to zero. Third, the symmetry property of the conceptual distance can be exploited such that $\mathbf{X}_{i,j} = \mathbf{X}_{j,i}$. The **Clustering** function can be implemented by either a partitional clustering or a hierarchical clustering [83]. The **SelectFeatureFromClust** function can be implemented in multiple ways. One implementation using $\mathbf{M}$ can be done by selecting a feature whose the conceptual distance to other features in the cluster is the shortest. Another option uses features' data by selecting a feature whose data is closest to the average or median data of all features within the cluster, which can help smooth out noise among these features. Combining these two sources of information is another way to select a representation for each cluster, which may lead to a better chosen subset.

Algorithm 3: Choosing the set of k non-redundant features

input: $\mathcal{L}$ = a set of input features' labels
k = number of clusters
O = SPARQL endpoint of an ontology
R = a symbol denoting the inheritance relation
T = a symbol denoting $\top$
$\mathbf{X}$ = a set of input features

Output: $\mathbf{X}_k$ = a set of k non-redundant features

/* This algorithm use Algorithm 1 to compute ϱ denoted by $\mathbf{SPARQL}_{\varrho}$ */

$\mathbf{X}_k \leftarrow \emptyset$
$\mathbf{M} \leftarrow []$
/* Compute the distance at the conceptual level for each pair of input features */
foreach $\ell_i \in \mathcal{L}$ **do**
  $\mathbf{m} \leftarrow []$
  $C_i \leftarrow$ **FindConcepts**(ℓ_i, O)
  foreach $\ell_j \in \mathcal{L}$ **do**
    $C_j \leftarrow$ **FindConcepts**(ℓ_j, O)
    $Q \leftarrow []$
    foreach $U \in C_i$ **do**
      foreach $V \in C_j$ **do**
        $s_{ont} \leftarrow \mathbf{SPARQL}_{\varrho}(U, V, R, T)$
        add s_{ont} into Q
    $s'_{ont} \leftarrow$ **max**(Q)
    add $(1 - s'_{ont})$ into $\mathbf{m}$
  add $\mathbf{m}$ into $\mathbf{M}$
/* Clustering features using $\mathbf{M}$ */
clusts $\leftarrow \mathbf{Clustering}(\mathbf{M}, k)$
/* Selecting a feature from each cluster */
foreach clust $\in$ clusts **do**
  $X \leftarrow \mathbf{SelectFeatureFromClust}(\text{clust}, \mathbf{M}, \mathbf{X})$
  add X into $\mathbf{X}_k$
return $\mathbf{X}_k$

4.4.3 Experimental Results

This experiment aimed to demonstrate the effect of parameter k on the clusters to be formed and the quality of a representation to be selected from each cluster. Algorithm 3 was implemented with DBpedia, as a source to compute the conceptual distance between two input features. The standard agglomerative clustering with the average linkage criterion [83] was used to produce k clusters from the distance matrix $\mathbf{M}$. Three different values of k were assigned to depict three different levels of sparsity and potential information loss of a selected feature subset: high (k = 5), medium (k = 20), and low (k = 40). For the purpose of visualization, Principal Component Analysis (PCA) [20] was applied to reduce the dimensions of the matrix $\mathbf{M}$ to two dimensions, the clustering algorithm is then performed on this reduced $\mathbf{M}$. The clusters of features constructed with three different values of k are presented in Figure 4.11, Figure 4.12, and Figures 4.13, one for each value.

These three figures show the clusters of features at the conceptual level, and they are merely the two dimensional representations transformed from a complex feature embedding. We can see from Figure 4.11 that when the number of clusters k is too low, the input features are clumped together into a few large clusters. The features in a same cluster may not be redundant to one another, for instance two input features with label "political stability index" (11) and "average yearly temperature" (59) were in cluster number 3. Choosing a representative feature from a large cluster would incur information loss. On the other hand, if the number of clusters is too high as shown in Figure 4.13, some potential redundant features may not be clustered together. For example, the features labeled by "total budget expenditures" (25) and "purchasing power parity" (13) were in different clusters, even though one can be derived from the other. However, these two features are clustered together when k is 20. These results point out that a suitable number of clusters may depend on the number of redundant features in a given dataset. If the dataset contains many redundant features, setting k to a low value could be a wise choice in order to ensure that the redundant features are clustered together. In the case of the dataset has none or only few redundant features, using high k would reduce the chance of losing useful information for explaining the

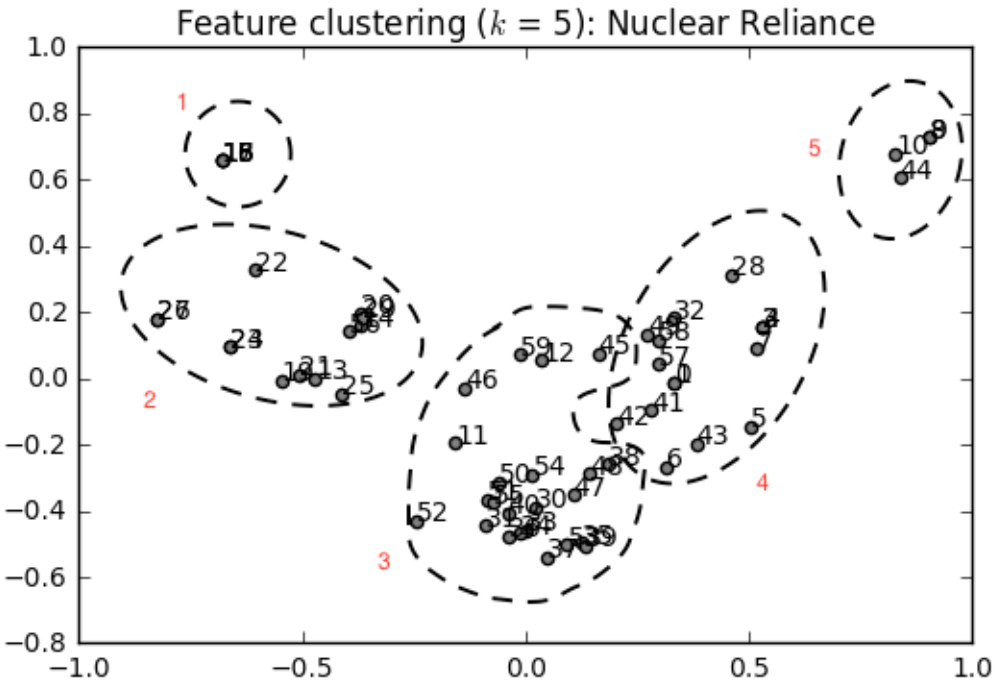

Figure 4.11: The figure shows five clusters of input features in *Nuclear Reliance* dataset. DBpedia is used to compute the conceptual distance between input features.

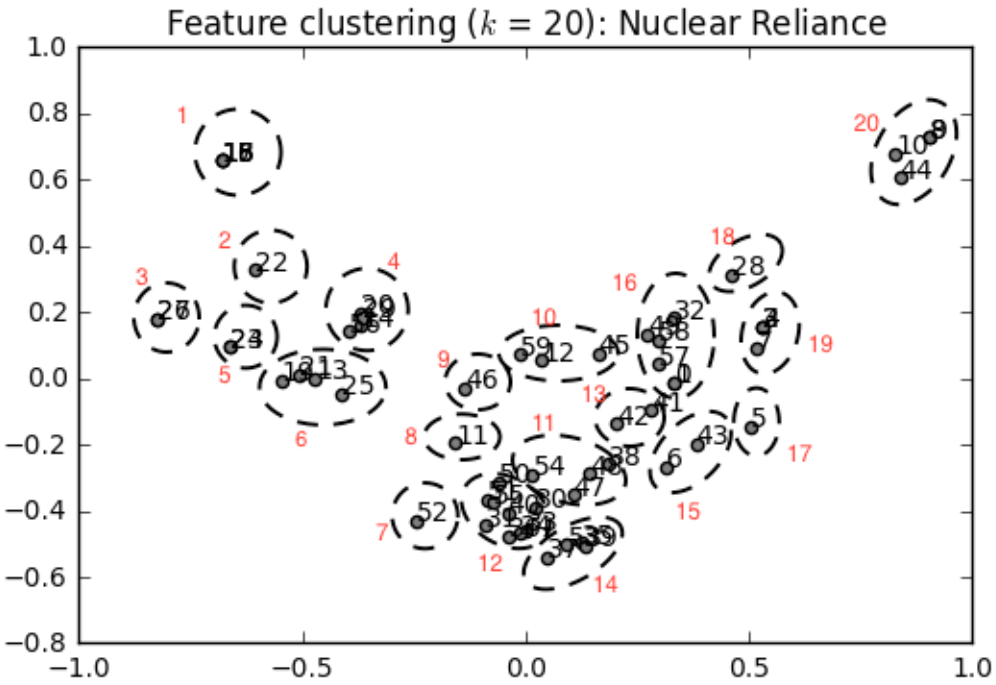

Figure 4.12: The figure shows twenty clusters of input features in *Nuclear Reliance* dataset. DBpedia is used to compute the conceptual distance between input features.

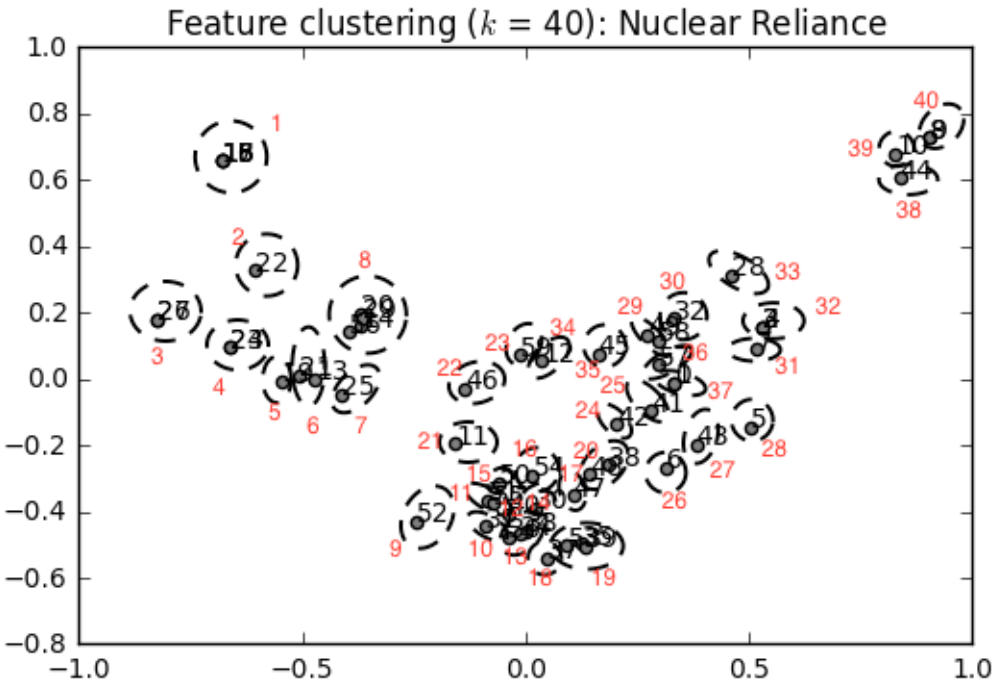

Figure 4.13: The figure shows forty clusters of input features in *Nuclear Reliance* dataset. DBpedia is used to compute the conceptual distance between input features.

variance of an output feature.

4.5 Feature Selection with Redundancy and Relevance Analyses

We introduce a feature selection method that examines both redundancy and relevance of input features at the conceptual level. Combining the relevance analysis (see Section 4.3) and redundancy analysis (see Section 4.4), this method chooses a feature subset containing features that are not redundant to each other and are relevant to the output feature. The architecture of this method is presented in Figure 4.14, in which the feature clustering algorithm as explained in Algorithm 3 is performed first to select a subset of k non-redundant features, $\mathbf{X}_k$, from a given original feature set. A total of p relevant features (where $p < k$) are then chosen from $\mathbf{X}_k$ using Algorithm 2, yielding a subset of p relevant features returned as the selected feature subset.

The idea underlying this feature selection method is that it creates an embedding space representing a conceptual organization of given features (both $\mathbf{X}$ and Y), in which each feature is placed

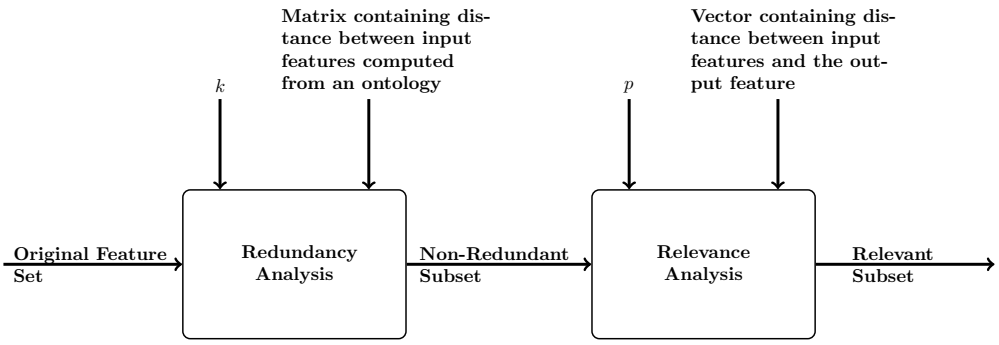

Figure 4.14: Following the overview in Figure 4.1, this figure shows an architecture of the proposed feature selection method, which can handle both feature redundancy and relevance.

in the embedding with respect to its conceptual distance to other features. An embedding of the input features is constructed first in order to perform the clustering via redundancy analysis, as shown in Figure 4.15, where the input features that are placed nearby in the space are clustered together. A feature whose conceptual distance to other features in the same cluster is the shortest is selected as a representation of the cluster. The output feature Y is then placed in the embedding space as shown in Figure 4.16. The p nearest input features to Y are then chosen from the k representative features, as a selected feature subset.

4.5.1 Experimental Results

The experimental results presented in Section 4.3 point out the usefulness and effectiveness of the conceptual distance computed from the ontologies in selecting relevant features. In this section, we aimed to study the impact of the redundancy analysis on the prediction accuracy of a model being built. There are four main questions to be addressed in this experiment. First, does combining the redundancy analysis with relevance analysis help in improving the generalization of a model compared to using only relevant analysis? That is we want to know whether or not removing some redundant features makes any difference. Second, how do the level of sparsity and information loss (the number of clusters) effect the prediction performance of a model? We expect that balancing between sparsity and information loss should yield a better model. Third, does redundancy analysis work across ontologies? That is we want to study the effect of different

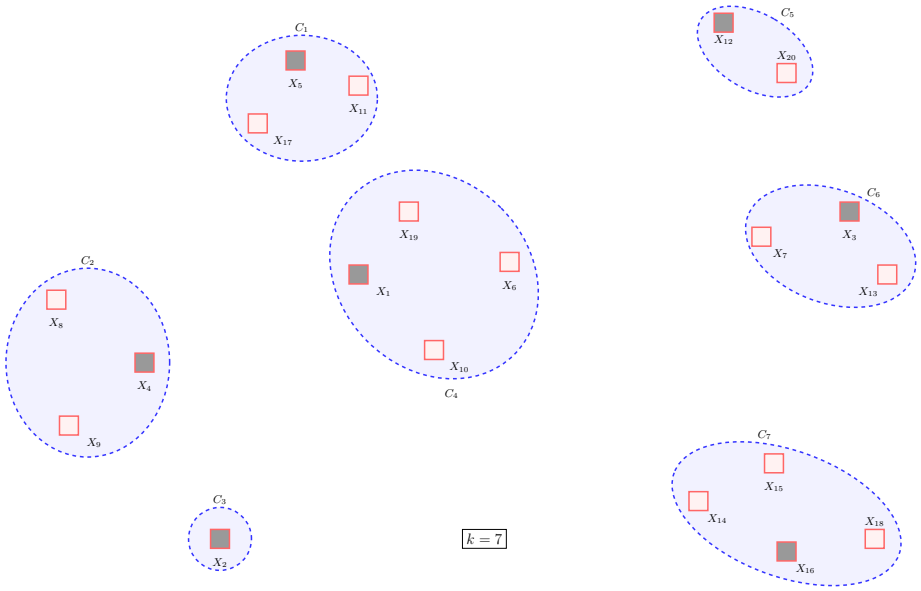

Figure 4.15: This figure shows the seven clusters (purple ovals) of input features (red rectangles) over a dummy embedding space are constructed via redundancy analysis. The gray-filled red rectangles represent the representations selected from the clusters.

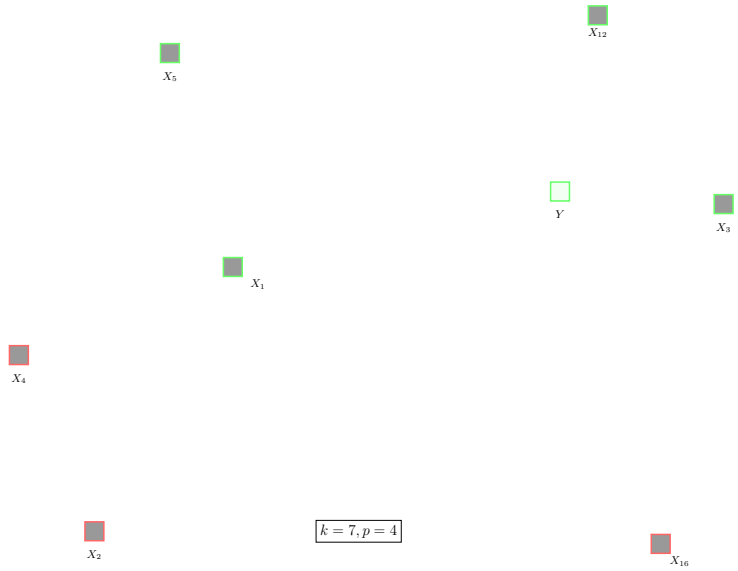

Figure 4.16: The relevance analysis is operated by placing the output feature (green rectangle) in the embedding space. The p nearest input features to Y are then selected from the k non-redundant feature subset (representations of the clusters)

category organizations in different ontologies on clustering and selecting non-redundant feature subset. Fourth, does redundancy analysis work across problem domains? We want to investigate the effect of removing redundant input features on the performance of model built from different datasets.

4.5.1.1 Does Redundancy Analysis Help?

In order to address this question, the feature selection method was tested with *Violent Crime*, as relevant analysis yields a good model on this dataset (see Section 4.3.5.2). In this experiment, the feature selection method was implemented with DBpedia (the results obtained from Yago will be shown in the following section). The parameter k was set to the medium level ($k = 20$), as we expect to get the best result. That is the relevant analysis that takes the output from the redundancy analysis will select p relevant features ($p < 20$) from 20 non-redundant features.

The *Violent Crime* dataset was split into training and test sets, in which the test set contains 30% of the examples. A linear model was learned from the training set using a standard linear regression method and its prediction accuracy was quantified on the test set via the residual sum of squares (RSS). The parameter p was incremented from 3 to 20 to specify the final number of input features to be selected (*i.e.,* sparsity level). For each specific p, the process of learning a model was repeated 50 times, the mean and standard deviation of RSS from 50 iterations were collected for the plots. The obtained results are shown in Figure 4.17 and were compared with the results obtained in Section 4.3.5.2 (see Figure 4.4), where only relevance analysis was used.

From Figure 4.17, we can see that the overall performance of models built from subsets selected by the feature selection method does not change much from those using only relevant analysis. This could be because of this dataset does not contain many redundant features. This method yields better model when p is low (obtaining the best model at $p = 7$) indicating that some relevant features but redundant to others were removed. Notice that the standard deviation of RSS is significantly reduced compared to that using only relevance analysis, which could be because of selecting only a representation for each cluster can reduce noise. The results point out the effectiveness and informativeness of the conceptual distance in clustering features and selecting representation, as

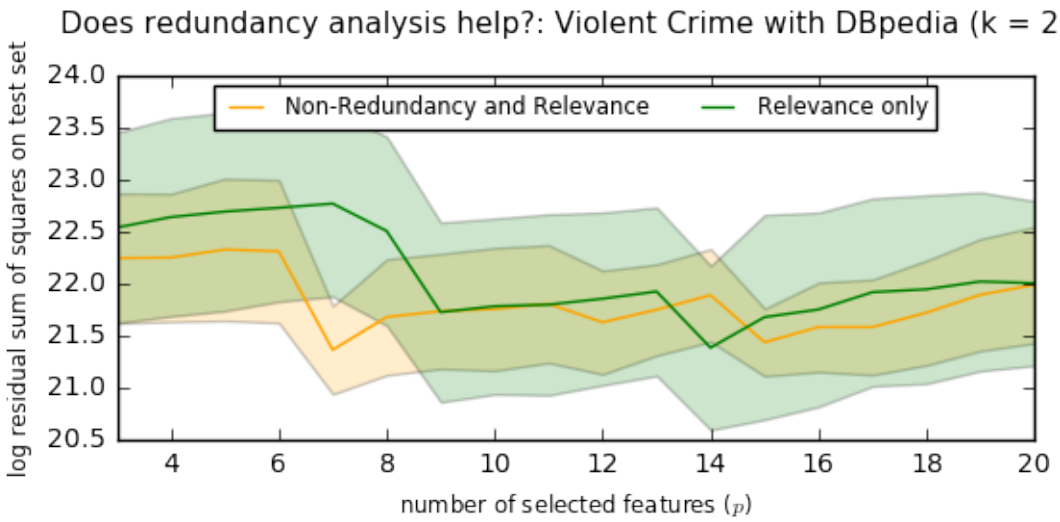

Figure 4.17: This figure compares the prediction accuracies of models built from non-redundancy and relevance subsets (orange line) with the subsets obtained from relevance analysis (green line)

the remaining features still yield a good model with more robust prediction performance.

4.5.1.2 Effect of Parameter k

In this section, the effect of the level of sparsity and information loss, as specified by the parameter k, on the performance of a model being built is studied. Two experiments similar to the former section were conducted, in which the parameter k of one experiment was set to the low value (k = 5) and the parameter k in another experiment was set to the high value (k = 40). The results of these experiments are shown in Figure 4.18 and Figure 4.19, one for each specific value of k. They were compared with the results obtained in Section 4.3.5.2 (see Figure 4.4), where only relevance analysis was used (the parameter p was extend to 40 from that figure).

Comparing results in Figure 4.18 with the results in Figure 4.17, where the level of sparsity and information loss is medium, surprisingly, the performance of the approach when k is low is comparable to that of k is medium. Deeper investigation needs in order to understand the reason underlying this behavior. Comparing results in Figure 4.19 with the results in Figure 4.18 and Figure 4.17, we can see that when the level of sparsity and information loss is too low, redundancy analysis does not help, as expected.

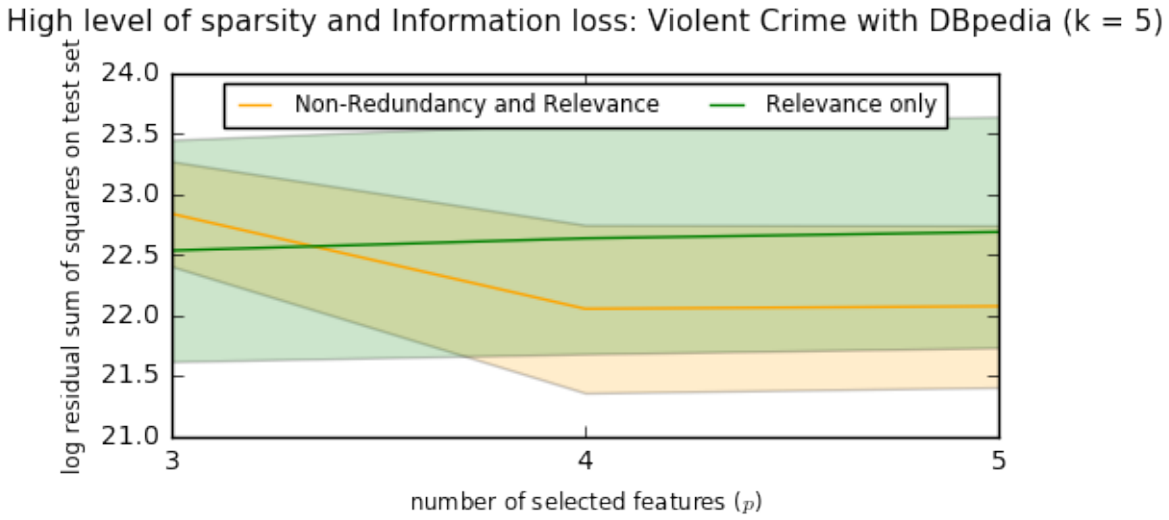

Figure 4.18: This figure compares the prediction accuracies of models built from non-redundancy and relevance subsets (orange line) with the subsets obtained from relevance analysis (green line). The level of sparsity and information loss is high, as k was set to 5

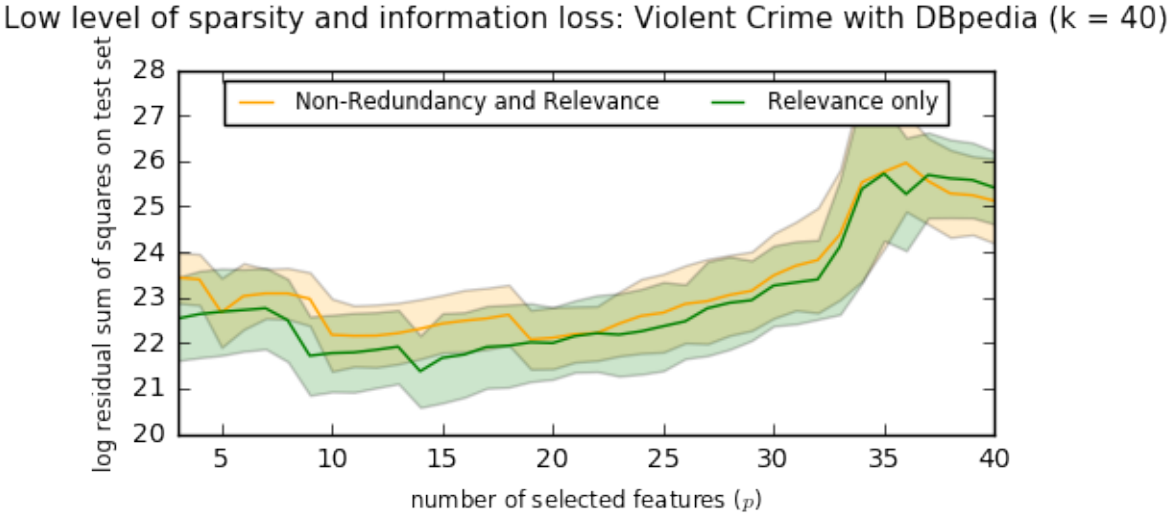

Figure 4.19: This figure compares the prediction accuracies of models built from non-redundancy and relevance subsets (orange line) with the subsets obtained from relevance analysis (green line). The level of sparsity and information loss is low, as k was set to 40

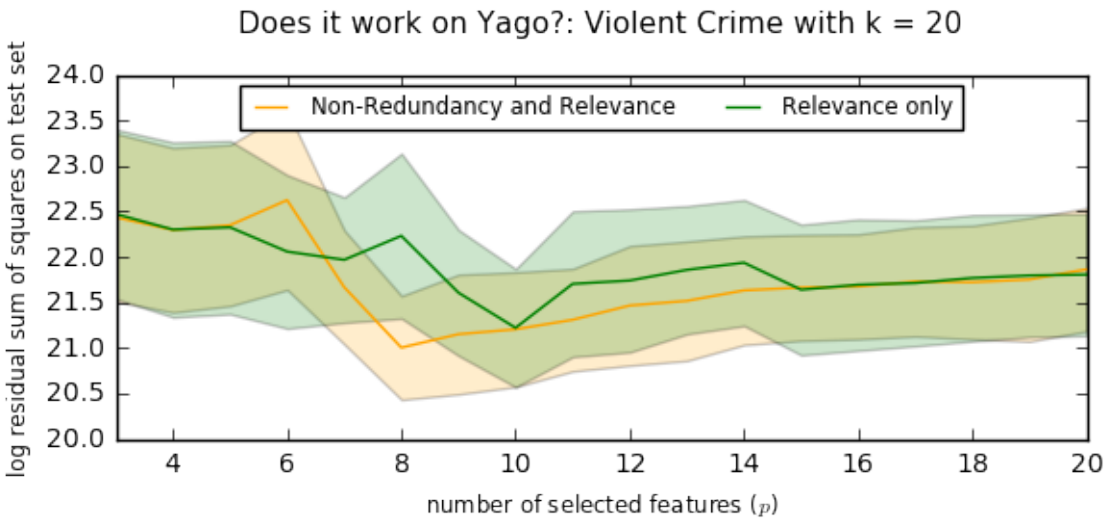

Figure 4.20: This figure compares the prediction accuracies of models built from non-redundancy and relevance subsets (orange line) with the subsets obtained from relevance analysis (green line) using Yago.

4.5.1.3 Does Our Approach Work Across Ontologies?

An experiment in this section aim to address the question concerning whether the ontological background knowledge, as used in our approach, is effective across ontologies. Despite the various organizations of the ontological categories or classes in different ontologies, we want to demonstrate that the general information used to compute the conceptual distance for redundancy analysis can be obtained effectively from other ontologies besides DBpedia. An experiment similar to the experiment in Section 4.5.1.1 was conducted except that Yago was used. If the conceptual distance used for redundancy analysis can be obtained effectively from Yago, we should see the behavior of our feature selection method similar to the plots in Figure 4.17.

Comparing the plots in Figure 4.20 with the plots in Figure 4.17, we can see that the behavior of our approach when using Yago does not change markedly from using DBpedia. These results support the belief that our approach works across ontologies and point out that both ontologies provide useful information to perform redundancy and relevance analyses.

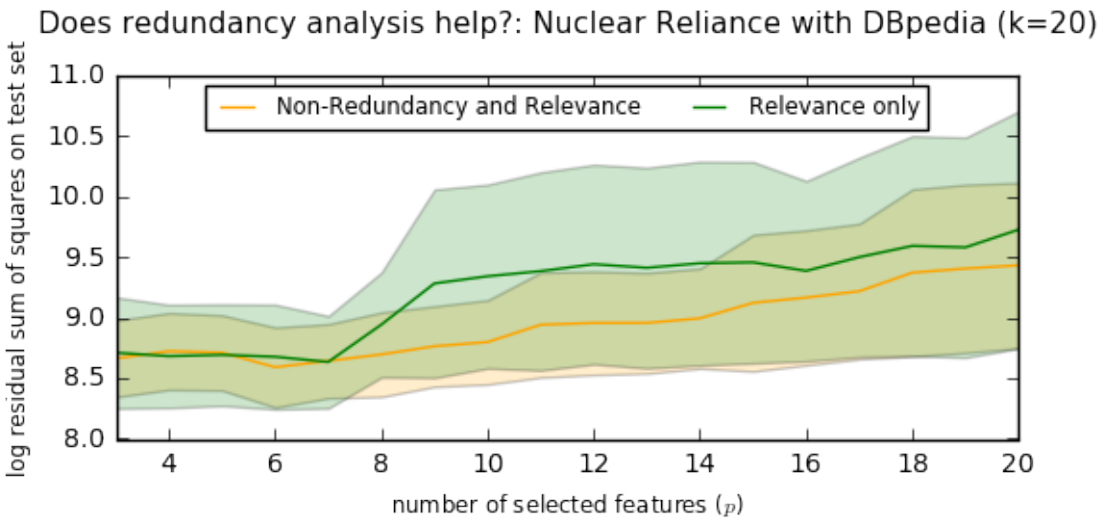

Figure 4.21: This figure compares the prediction accuracies of models built from non-redundancy and relevance subsets (orange line) with the subsets obtained from relevance analysis (green line) using DBpedia on the *Nuclear Reliance* dataset

4.5.1.4 Does Our Approach Work Across Problem Domains?

Experiments in this section aimed to address the question concerning the effect of redundancy analysis on the performance of models built for different problem domains. Two experiments similar to the experiments in Section 4.5.1.1 and Section 4.5.1.3 were conducted for both DBpedia and Yago except the dataset has changed to *Nuclear Reliance*. Based on the experimental results in Section 4.3.5.2 and Section 4.3.5.4, we can see that only relevance analysis does not do so well on the *Nuclear Reliance* dataset. The idea is that removing some redundant features may help produce better models than applying only relevance analysis on this dataset. The experimental results are shown in Figure 4.21 and Figure 4.22, one for each ontology. They were compared with the results obtained in Section 4.3.5.2 (see Figure 4.3) and Section 4.3.5.4 (see Figure 4.7), where only relevance analysis was used.

The plots from both figures show that the performance of our approach when adding redundancy analysis does not change markedly from that of using only relevant analysis on both ontologies. This could be because of the *Nuclear Reliance* dataset does not contain many redundant

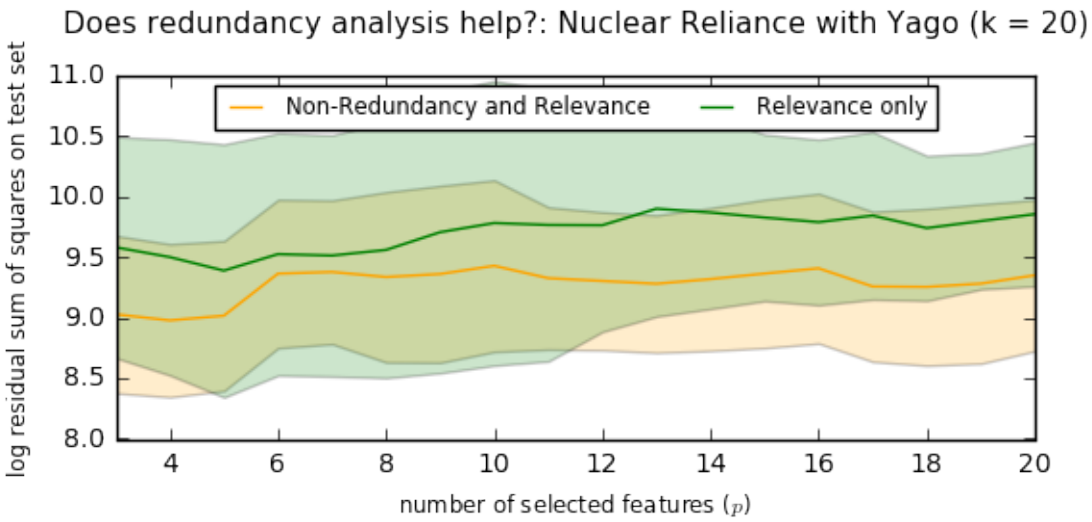

Figure 4.22: This figure compares the prediction accuracies of models built from non-redundancy and relevance subsets (orange line) with the subsets obtained from relevance analysis (green line) using Yago on the *Nuclear Reliance* dataset.

features. However, the models with more robust prediction performance were obtained on both ontologies, as shown by the significantly decreasing of RSS's standard deviation. The similar behaviors of our approach on both datasets indicate that it works across problem domains, where its efficiency on a particular domain may depend on the adequacy of high-level knowledge codified in the ontology and the nature of the particular dataset.

4.6 Summary

This chapter discussed how to use the conceptual distance computed from an ontology in feature selection. The criterion for evaluating feature relevance and redundancy are defined by using this ontological measurement so that semantic information of a feature is taken into account when evaluating the feature. The conceptual distance is shown to be useful and informative. Its effectiveness is comparable to the traditional data-driven measurements in choosing a feature subset for building a model. The feature selection method that makes use of the proposed criteria to handle both feature redundancy and relevance is described, and the experimental results point out that it seems to be the promising approach.

5. HELPING NOVICES TO BECOME DOMAIN EXPERTS

The experimental results from the last section demonstrate the usefulness and effectiveness of the conceptual distance computed from ontologies in semantically selecting relevant feature subset for learning models. This chapter aims to address the question of how to further use rich semantic information in an ontological structure to aid a novice in building a model more conveniently and accurately. For a novice, it usually comes down to two general difficulties when building a model: which data should be used, and where to find these data. In order to alleviate such difficulties, we developed a model building framework* that uses the intuition underlying the conceptual distance to retrieve a set of relevant concepts, and then extract data that associated with the concepts from available online data sources to construct a dataset for learning a model. Details of the proposed framework along with how to implement this framework with DBPedia and Wikipedia are explained. The experimental result shows that a model built from our framework performs comparably to the model built by domain experts.

5.1 Model Building Process Used by Experts

The model building framework developed in this dissertation is inspired by the modeling process employed by Nelson and Sprecher [1] to build a model of nuclear power usage in a given country to help understand civil nuclear proliferation. Figure 5.1 shows the key ideas distilled from their modeling process. As experts, they used their knowledge of the nuclear domain to identify factors that are relevant to nuclear power usage. Critically, the choices made regarding the data inputs for the model were directed by knowledge at the conceptual level rather than correlations in the data themselves. Moreover, most of the data that was used to build their model came from existing online data sources, such as government and public organization websites. While this modeling process had broad promise, it involves two challenges for a non-expert. Firstly, it relies

*Reprinted with permission from "Helping Novices Avoid the Hazards of Data: Leveraging Ontologies to Improve Model Generalization Automatically with Online Data Sources", by Sasin Janpuangtong and Dylan Shell, 2016, AI Magazine, vol. 37, no. 2, pp. 19-32, Copyright 2016 by Association for the Advancement of Artificial Intelligence.

heavily on knowledge from the person building the model. Secondly, the data collection itself was performed manually. Better data collection methods must be developed to help anyone collect data from different sources across a variety of formats.

5.2 Model Building Process with an Ontology

We have set out to develop a semi-automated model building framework [84] that adopts key ideas from, but also addresses the challenges of, the modeling process described above.[1] The framework operates as follows:

(i) An existing ontology is used as a source of background knowledge rather than relying on knowledge from the person building a model.

(ii) Since ontologies are precise machine manipulable representation of *a priori* structured relationships among concepts in a problem domain, they also enable a machine to explore the knowledge in an ordered manner to determine the relevance of various concepts. Such concepts are then used to construct a hypothesis space, and data are used to find the best model in this space using a learning method.

(iii) Operationalization of an ontological concept to corresponding measurements can be done by finding a data source corresponding to that concept, then looking for measurements from information published on that data source.

(iv) A data extraction component is introduced into the framework to help a novice extract desired data from a target data source easily.

5.2.1 Model Building Framework

The framework is illustrated in Figure 5.2. It is an end-to-end approach that employs knowledge at two levels, *high-level concepts* and their relationships in an ontology, and *low-level data* from existing data sources. Model building proceeds from left to right and consists of three main

[1] Although we have emphasized its use by Nelson and Sprecher, the approach represents a standard approach in several sciences.

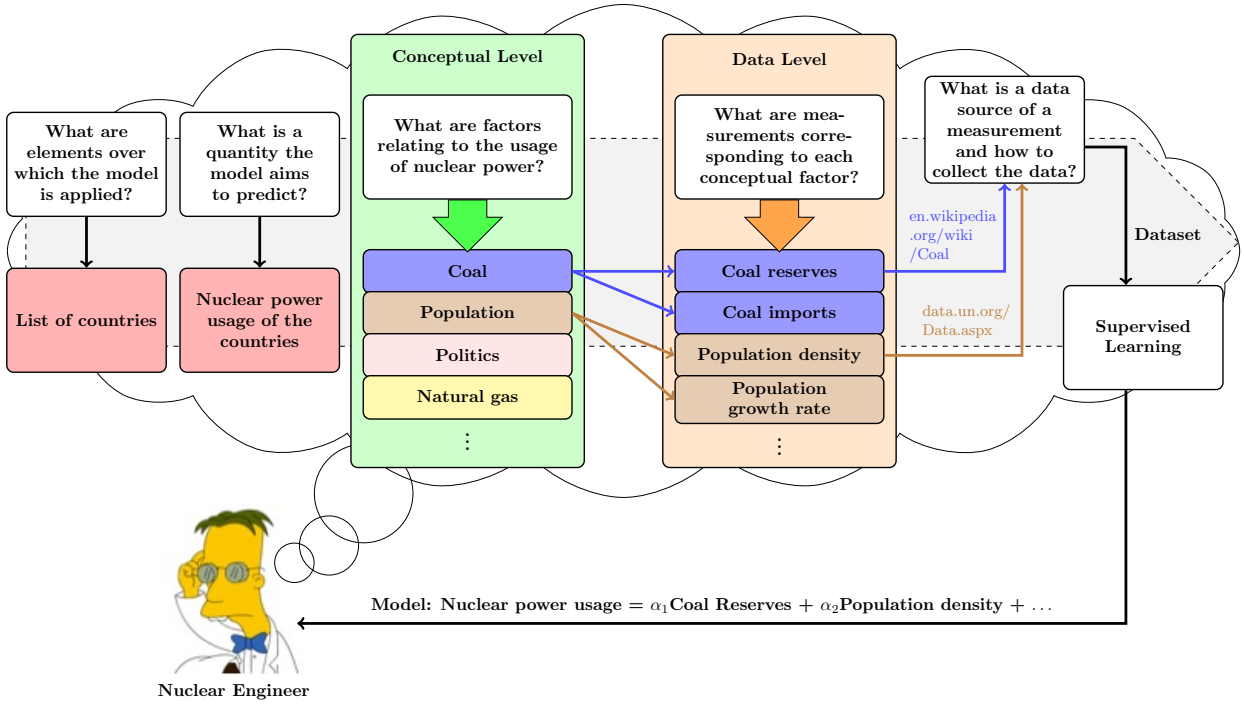

Figure 5.1: Modeling process extracted from Nelson and Sprecher [1]. The experts carry out the process progressing from left to right to build a model for predicting nuclear power use. The expert starts by specifying a list of countries and then collecting nuclear power data of those countries from an existing data source. Background knowledge is used to come up with several hypotheses about the factors that may influence nuclear power usage. For example, a country with a large coal resource might be expected to use coal to produce electricity rather than using nuclear power. Thus, coal may affect nuclear power usage. Next, the expert moves to find measurements that correspond with the factors. For instance, coal reserves and coal imports are possible measurements relating to the coal in a country. The expert selects one of these measurements for each factor and uses a data source to provide data for the selected measurement. The collected data from these sources are used to construct a dataset for learning. Finally, the expert uses the resultant dataset with a designated learning method to build a model.

components. Each component is associated with a corresponding outside component to perform its task. For example, using this framework to build a model for predicting nuclear power usage of countries, a user starts by giving the first component a query string *"Nuclear power"* that will be the output of the model, representing the quantity he aims to predict. This component uses structured relationships in an ontology (which we assume is given) to automatically retrieve a set of concepts that are relevant to the input query, which we denote Θ.

So far, relationships are only captured at a high-level between concepts. To evaluate its predictive value, concepts in Θ must be operationalized to corresponding data. To perform this step the user describes the set of elements over which the model is applied (the model's domain), and what the model aims to predict (the model's output). Tabular input containing two columns is used for this purpose. In the example, the first column contains list of countries and the second column contains nuclear power usage data. We assume that the user already has this tabular input (*e.g.*, it could be collected from a web page publishing these data in tabular format). The input query and tabular input are used to form a question, in this case essentially asking *"Which attributes of countries are relevant to nuclear power?"* This tabular input is also stored in the framework as an initial dataset.

For each concept in Θ, a data source (*e.g.* a webpage, excel file, *etc.*) that provides measurements or values is specified. The user selects a suitable scraping module (*e.g.*, table scraping, list scraping) to extract contents. If one of the concepts is *"Coal"*, representing the energy resource, then the user may provide the Wikipedia article *http://en.wikipedia.org/wiki/Coal*, which contains several tables with data related to this concept. He selects the table scraping tool from the framework. The second component accesses the article and uses the selected scraper and data from the tabular input to extract all tables containing data about countries. These tables are represented as possible measurements related to the coal concept. The user chooses a table (the one providing coal reserves), which has six columns (*e.g.* SubBituminous, Lignite, Total, *etc.*) containing data related to different aspects of coal reserves. He selects one of these column. Data from the selected column are extracted and then added into the dataset. This step is repeated until all concepts in

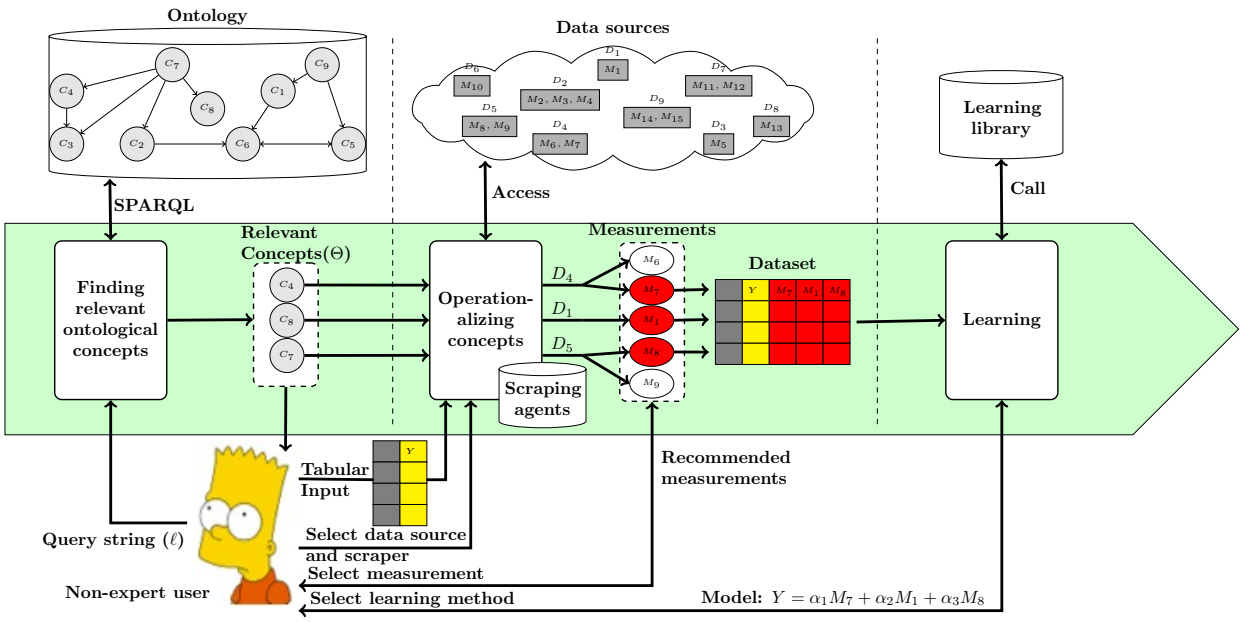

Figure 5.2: Proposed learning framework. This framework incorporates both an existing ontology and data sources to build a model from data. Knowledge in the ontology is used to construct an initial model that is composed of relevant ontological concepts and data, the latter being associated with the concepts and retrieved from existing data sources. The model is built and validated on the data using a selected learning method.

Θ are operationalized to data and added to the dataset. At this point, the dataset is ready for use in the learning process (*i.e.*, the third component). The user selects a learning method (*e.g.*, linear regression, decision tree). The framework calls the selected method from an existing library (*e.g.* scikit-learn, R) to build a model from the dataset. The model is returned to the user so it can be used for predictions, and to determine (*e.g.*, from coefficients) the relative importance of the various concepts. The user might conclude, for instance, that large coal reserves reduce the likelihood of a country building a nuclear power station.

The framework exploits an existing relationship between three components: concepts in an ontology, existing data sources, and measurements of the concept and the given examples. Next we emphasize how the components involved help minimize the human effort required.

5.3 Finding Relevant Concepts from an Ontology

Given a query string and an ontology $\mathcal{O}$ containing knowledge encoded within an RDF data model, we retrieve and rank concepts from $\mathcal{O}$ by using its category organization, which is a broadly applicable knowledge representation, useful across many ontologies. This organization is typically composed of two main elements: categories and concepts, in which closely related concepts are organized in the same category. Closely related specific categories are also organized into the same broader category forming super- and sub-categories relationships. We use SPARQL, the query language for RDF data, queries to acquire knowledge from the category organization described in $\mathcal{O}$ without any preprocessing effort. The $\mathcal{O}$'s SPARQL endpoint is denoted by O

5.3.1 Hyperlink Induced Topic Search

We adopt the ideas of the Hyperlink Induced Topic Search (HITS) algorithm [85], originally for searching and ranking relevant web documents on a given topic by considering two different notions of relevance: hubs and authorities. Representing ontological categories as hubs and concepts as authorities, HITS can be employed on the ontology to find and rank relevant concepts for a given input query. The intuition underlying our algorithm is that if a category is relevant to the input query, then all concepts in that category should also be relevant to the query. Likewise, if a concept is relevant to the query then all categories containing the concept should be considered as relevant as well. Since an ontological element is explicitly defined either as a category or concept, it has only one score associated with it (*i.e.,* the hub score for a category and the authority score for a concept), which also means that the scoring computation from HITS can avoid iterative updates.

5.3.2 Scoring Schemes

Two separate scoring schemes are used to quantify the relevance and utility of finding other related information for categories and concepts. Both schemes are the product of two components: voting and frequency. Voting captures the relationships between categories and concepts, while frequency tracks how often categories or concepts reappear during execution of the algorithm.

5.3.2.1 *Category Scoring*

A category scores well if it links to many rare concepts that are relevant to an input query. Thus, category *c*'s score comes from voting from concepts in *c* and how many relevant concepts appear in *c* so that

$$\text{Score}(c) = \sum_{i \in I_R} \text{IF}(i, c) \times \sum_{i \in I_R} \text{IVote}(i, c), \tag{5.1}$$

where I_R denotes a set of relevant concepts, IF(i, c) is a function with value 1 if a concept $i \in c$ or otherwise 0. IVote(i, c) gives the vote based on rarity of a concept *i*, such that

$$\text{IVote}(i, c) = \begin{cases} 1/|\text{categories containing } i| & \text{if} \quad i \in c, \\ 0 & \text{otherwise.} \end{cases}$$

The IVote function states that if *i* appears in many categories, it is not a rare concept and its vote is shared among the many categories, so a category containing many common concepts is penalized.

5.3.2.2 *Concept Scoring*

A concept scores well if it is linked by many relevant categories. Thus, concept *i*'s score comes from the votes of categories containing *i*, and how many relevant categories *i* appears in, such that

$$\text{Score}(i) = \sum_{c \in C_R} \text{CF}(i, c) \times \sum_{c \in C_R} \text{CVote}(i, c), \tag{5.2}$$

where C_R denotes a set of relevant categories, CF(i, c) is 1 if $i \in$ category *c* or otherwise 0, and CVote(i, c) returns the score of *c* as:

$$\text{CVote}(i, c) = \begin{cases} \text{Score}(c) & \text{if} \quad i \in c, \\ 0 & \text{otherwise.} \end{cases}$$

5.3.3 Algorithm Details

The algorithm performs three steps to find and rank concepts that are relevant to the input query ℓ in the category organization. Details appear in Algorithm 4. The algorithm starts by finding a concept q, whose label matches (textually) ℓ, to represent the query. If more than one concept is found, the first is selected. Next, all concepts that link to (inlinks of) or receive a link from (outlinks of) q are retrieved from $\mathcal{O}$ to construct a set of initial relevant concepts, I_R.

Finding a set of relevant categories is performed in the second step (lines 5–24). For each concept in I_R, its categories are discovered and scores calculated using Equation (5.1). The categories are sorted with respect to their score and the top *n* categories are selected. Heuristics are used to further select categories from these top categories, finding those that

- contain few concepts (*i.e.*, discarding categories which list names of films, animals, or scientists),
- contain neither very few nor many sub-categories (*i.e.*, categories that are too specific or too general). The resulting set is denoted by C_R

Input parameters *m*, *min*, and *max* are used to adjust this behavior. Suitable values depend on the ontology and problem domain.

The final step (lines 25–34) retrieves all concepts from each category in C_R, scoring each with Equation (5.2). All concepts are sorted by score before being returned as the output.

5.4 Implementation: DBpedia and Wikipedia

Our implementation leverages an existing ontology and online data sources; some details are worth discussing. We used DBpedia [39], the ontology counterpart of Wikipedia, as a source of background knowledge and used Wikipedia articles as data sources for corresponding DBpedia concepts. The vast amount of general knowledge in this ontology allowed testing on multiple case studies. Moreover, each DBpedia concept has a corresponding Wikipedia article often containing detailed information associated with that concept in different formats, such as text, list, or table.

Algorithm 4: Find and rank relevant ontological concepts

input: ℓ = Query string from user
O = Ontology (URL to its SPARQL endpoint)
n = Number of categories
m = Maximum concepts in category
min_sub = Minimum sub-categories in category
max_sub = Maximum sub-categories in category

Output: Θ = A set of ranked relevant concepts

```
/* All functions in this algorithm except SORT can be
   implemented by using only SPARQL.                    */
```

$q \leftarrow$ **getConceptFromStr**(ℓ, O)
$Out \leftarrow$ **FindOutLink**(q, O)
$In \leftarrow$ **FindInLink**(q, O)
$I_R \leftarrow q \cup Out \cup In$
foreach $i \in I_R$ **do**
  $Cats \leftarrow$ **getCategories**(i, O)
  $vote \leftarrow 1/Length(Cats)$
  foreach $c \in Cats$ **do**
    $C_{vote}[c] \leftarrow C_{vote}[c] + vote$
    $C_f[c] \leftarrow C_f[c] + 1$
foreach $c \in C_{vote}$ **do**
  $C_{score}[c] \leftarrow C_{vote}[c] \times C_f[c]$
$C_{score\_sort} \leftarrow$ **SORT**(C_{score})
$k \leftarrow 0$
while $k < t$ **do**
  $c \leftarrow C_{score\_sort}[k]$
  $mems \leftarrow$ **CountConcepts**(c, O)
  $subs \leftarrow$ **CountSubCategories**(c, O)
  if $mems < m \wedge (subs > min\_sub \wedge subs < max\_subs)$ **then**
    Add c into C_R
  $k \leftarrow k + 1$
foreach $i \in I_R$ **do**
  $I_f[i] \leftarrow 1$
foreach $c \in C_R$ **do**
  $Cons \leftarrow$ **getConcepts**(c, O)
  foreach $i \in Cons$ **do**
    $I_{vote}[i] \leftarrow I_{vote}[i] + C_{score}[c]$
    $I_f[i] \leftarrow I_f[i] + 1$
foreach $i \in I_{vote}$ **do**
  $I_{score}[i] \leftarrow I_{vote}[i] \times I_f[i]$
$\Theta \leftarrow$ **SORT**(I_{score})
Return Θ

Our implementation exploits this connection to automatically get an article of a concept and use it as a primary data source for finding possible measurements of the concept. Thus, the user does not need to specify a data source for a concept, allowing the system to build a model with minimal human effort (as highlighted in the sequence in Figure 5.3). Our system focuses on data that are published in tabular format since each table often encapsulates a complete, non-redundant set of facts [86], and tables structure data for easy automatic interpretation and extraction.

As shown in Figure 5.3, suppose the user would like to build a model for predicting GDP of countries. He can use this system by giving a query string *"Gross Domestic Product"*. The system retrieves relevant concepts from DBpedia. In the second step he selects a tabular input, with the first column containing a list of countries and the second column containing GDP data for each country. Then the system requests HTML data of Wikipedia articles corresponding to concepts output from the first step. The system processes the HTML data of these articles to construct a dataset. In the third step, the user selects a learning method. The system calls that method from Scikit-learn library to build a model from the dataset. We can see that the user gives the system only three inputs and then lets the system carry out the remainder of the steps to build a model. This meshes with our earlier motivation of producing a system to help the non-expert user to build a model from existing data easily. Implementation details of the first two steps are described in the following sections.

5.4.1 Finding Relevant Concepts from DBpedia

There are three important points when implementing Algorithm 1 with DBpedia. Firstly, internal links among Wikipedia articles are used to find inlinks and outlinks of concept ϵ. DBpedia already contains these internal links as RDF triples via the *wikiPageWikiLink* predicate. The intuition is that a human has placed these hyperlinks between articles as the conferral of authority from one article to others. Such an authorized article supplies related and additional information about an article that links to it. That is, this predicate indicates the relatedness between the corresponding concepts of the articles. Secondly, suitable values parameters *n* and *m* for DBpedia were found to be in the ranges 200–250 and 120–200, respectively, depending on the problem domain. While,

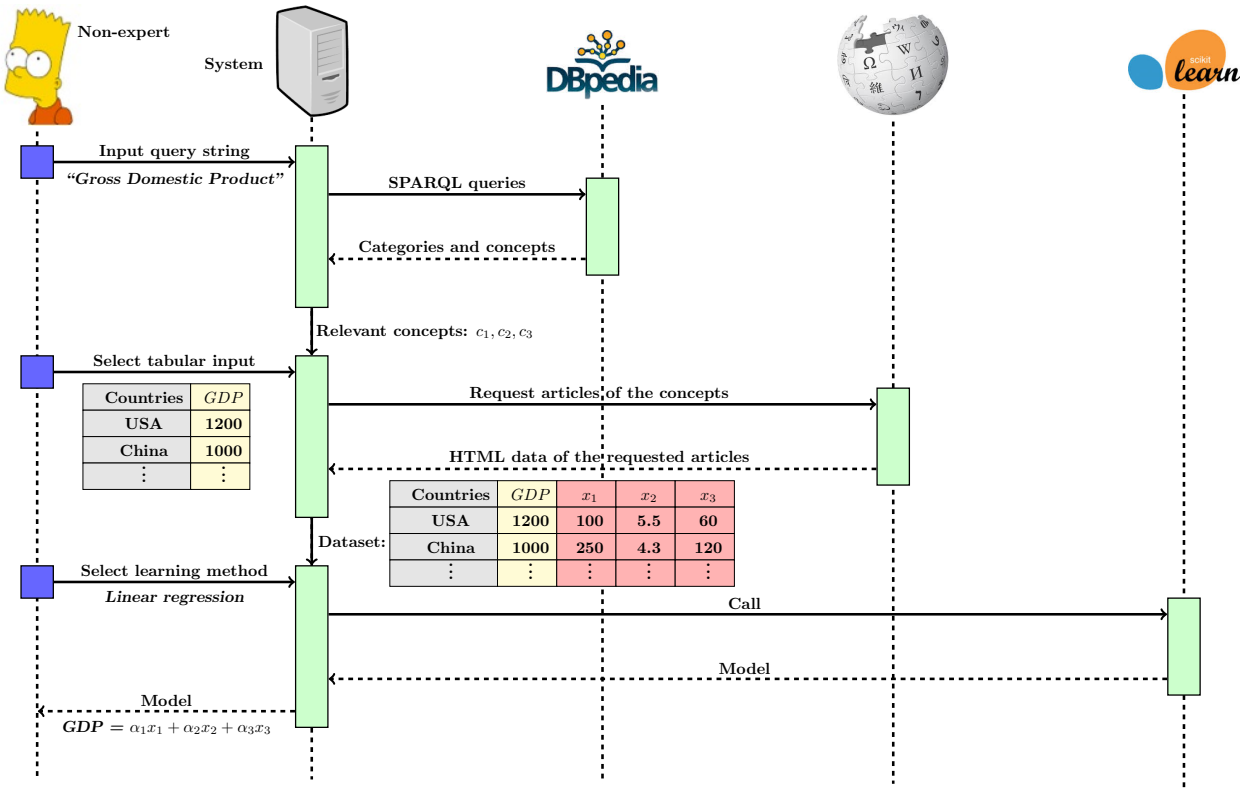

Figure 5.3: Sequence Diagram of the Implementation. A diagram showing an interaction use case with our DBpedia and Wikipedia implementation. It shows how a model can be built with little human effort. Light (green) rectangles indicate operations that are done automatically and dark (blue/purple) ones show where user intervention is required.

the values of *min* and *max* were set to 6 and 30, respectively, so that the DBpedia categories that are not too specific or too broad are retrieved. Lastly, we added a DBpedia specific condition at the end of the algorithm to further select only concepts whose name (after removing all prefixes) starts with the term *List_of*. We found that corresponding articles for these concepts usually have tables providing data about a specific aspect of the concept. Focusing on these types of concept allows our implementation to automatically find and collect data as needed. For instance, the Wikipedia article *List_of_countries_by_GDP* has a table that contains data for GDP by country (suiting the preceding example). Figure 5.4 shows how this implementation works to find relevant concepts from DBpedia for an input query. The sample results obtained via this implementation using input queries: *Poverty* and *Gross Domestic Product* are shown in Figure 5.5.

5.4.2 Collecting Data from Wikipedia Tables

Algorithm 5 automatically extracts data from a table on a Wikipedia page. For each concept in Θ, this algorithm starts by eliminating the concept that is redundant with q by checking whether or not the string ℓ appears in the concept's label. It then acquires the URL of a Wikipedia article associated with the concept from DBpedia, requests the article in HTML, and extracts all tables from the result. Heuristics are used to select a table that

- has one column, we call an "example column," that partially matches to elements in the first column of the tabular input (countries in the nuclear example)
- has this "example column" appearing in the first or second column, which indicates that the data published in this table are about the model's domain.

If no table is selected, the concept is discarded. The algorithm then seeks columns in the selected table that contain numerical data. The numerical column closest to the example column is selected; if no numerical column is found, the example column is used to construct a new column that contains binary data (*i.e.*, value 1 is assigned to indicate that an element from the first column of the tabular input appears in the example column, otherwise value 0 is assigned).

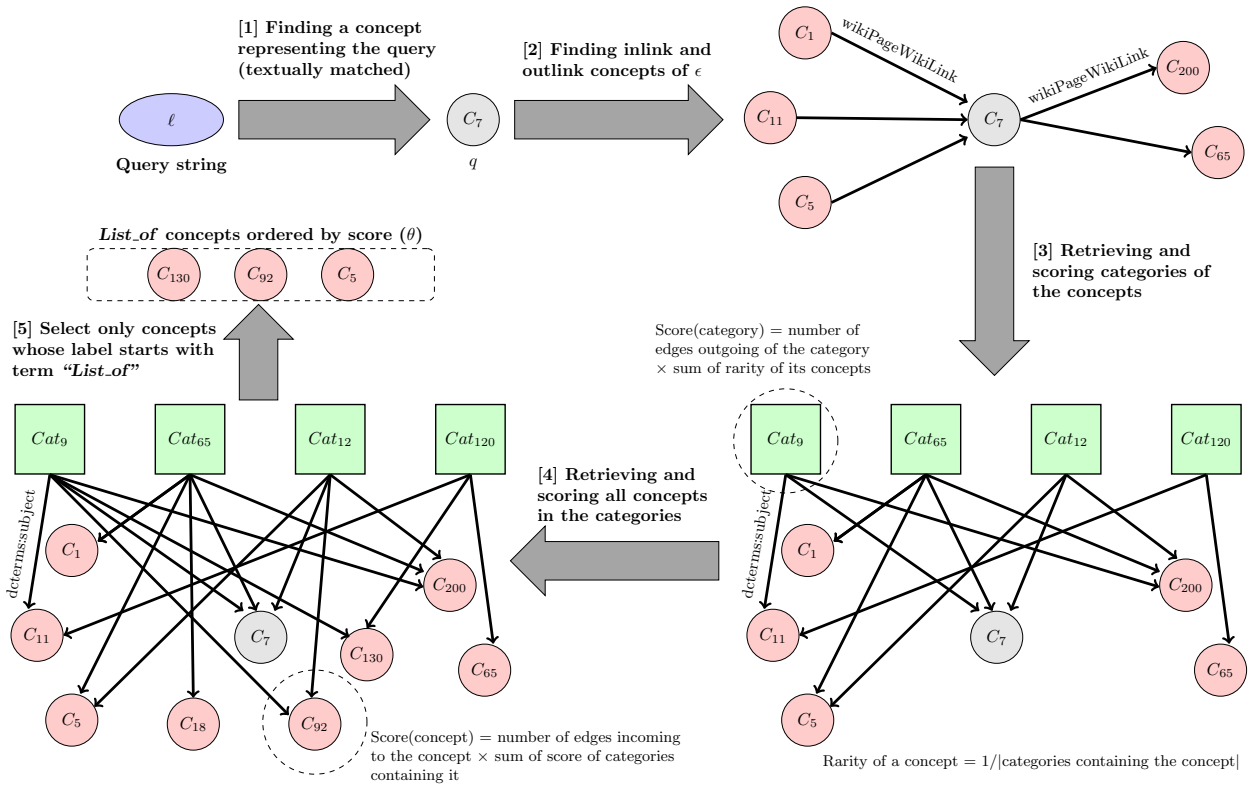

Figure 5.4: Steps of running Algorithm 4 with DBpedia. Running each step of the Algorithm 1 with taxonomic knowledge in DBpedia grows the graph as illustrated: [1] Finding a seed node, q representing the input query, ℓ. [2] Outlinks and inlinks of q (red filled circles) are added to the graph. [3] Categories of each added concepts are discovered. Relevant categories (filled rectangles) are selected to add to the graph. [4] Extra concepts from the added categories are retrieved and then added. [5] Concepts whose label starts with term *"List_of"* are selected and then returned as output.

ℓ = ***"Poverty"***		ℓ = ***"Gross Domestic Product"***
countries_by_percentage_of_population_living_in_poverty		Australian_states_and_territories_by_gross_state_product
countries_by_unemployment_rate		sovereign_states_by_external_assets
countries_by_employment_rate		countries_by_economic_freedom
countries_by_Sen_social_welfare_function		freedom_indices
permaculture_project		countries_by_Sen_social_welfare_function
sovereign_states_and_dependent_territories_by_fertility_rate		countries_by_percentage_of_population_living_in_poverty
global_manpower_fit_for_military_service		countries_by_percentage_of_population_suffering_from_undernourishment
wars_and_anthropogenic_disasters_by_death_toll		countries_by_energy_intensity
countries_by_sex_ratio		countries_by_future_gross_government_debt
countries_by_infant_mortality_rate		countries_by_public_debt

Figure 5.5: Top 10 *"List_of"* concepts of input queries: *"Poverty"* and *"Gross Domestic Product"*.

Algorithm 5: Extracting data from a table on Wikipedia article

input: Θ = a set of DBPedia concepts that their name starts with the term *List_of*
ℓ = The query string from the Algorithm 1
O = URL to SPARQL endpoint of DBPedia
t_0 = name of examples and output values in tabular format

Output: A dataset t in tabular format

```
1  t ← t_0
2  foreach i ∈ Θ do
3      if Contain(i, ℓ) then
4          continue
5      t_best ← null
6      ex_index ← 100
7      p ← GetWikiArticle(i, O)
8      T ← ExtractTables(p)
9      if T = null then
10         continue
11     foreach τ ∈ T do
12         c_index ← GetExamplesColumn(τ, t_0[0])
13         if c_index = −1 then
14             continue
15         if c_index < ex_index then
16             t_best ← τ
17             ex_index ← c_index
18     if t_best = null then
19         continue
20     col_new ← null
21     num_index ← FindNumericalColumn(t_best, ex_index)
22     if num_index = −1 then
23         col_new ← BiColumn(t_best[ex_index], t_0[0])
24     else
25         col_new ← NumColumn(t_best, ex_index, num_index, t_0[0])
26     t ← AddColumn(t, col_new)
27 Return t
```

At the end of this step, the system produces a tabular dataset containing data from attributes of elements over which the model is applied (countries in the nuclear example) and these attributes are also relevant to the input query.

Even though Algorithm 5 can automate table detection and data scraping of a Wikipedia article to operationalize a concept, this heuristic approach may select the wrong table or column to retrieve data yielding a poor final outcome. In a manual approach, on the other hand, the system only retrieves tables from an article and shows them to a user. The user then selects a table and a column to add to a dataset. Doing so, however, relies on the user having enough knowledge to select the correct table and column. A hybrid approach could be the better option, where the system initially chooses a table and a column for users and then lets them decide whether to accept that choice or change to another more appropriate table and/or column. The system can also provide a score for each table and/or column based on its contents (*e.g.,* number of matched countries in a table) to help the user make a wise choice.

5.4.3 Implementing the Framework with Other Ontologies and Data Sources

It is worth discussing the implementation of the proposed framework with ontologies and data sources other than DBpedia and Wikipedia. Firstly, in Algorithm 4 we assume that there is a concept, q, whose label matches an input query textually. An ontology used by the user could contain terminology that is more variable (*e.g.,* different acronyms and synonyms that do not directly match an input query). An additional step that can either automatically enrich the input query with synonyms or acronyms or find concepts that are semantically related to the query [87, 88] should precede the algorithm to resolve this issue. The user first selects one of concepts returning from this step and uses it as q. Secondly, other common predicates, such as *owl:sameAs* or *rdfs:seeAlso* could be used to find inlinks and outlinks of q rather than using *wikiPageWikiLink* predicate, which is specific to DBpedia. Thirdly, for Algorithm 5, choosing a corresponding data source for a concept can be done directly by a user (*e.g.,* an Excel file containing data corresponding to the concept). A suitable user interface should be supplied to allow a user to specify the location of that data source. A search engine service, such as Google search service, could be employed in

the system to help a user easily find a suitable data source from the Internet. By preprocessing a label of a concept and then input it as a query to the search service, the user can get locations of online data sources corresponding to the concept and then select one of them to retrieve the data. Lastly, scrapers for different data formats, such as lists, texts, files, should be added to the data scraping toolbox, along with an interface that allows a user to select a suitable scraper. Inconsistent or poorly structured and formatted data pose difficulties for scraping. One way to handle this problem is to have the data scraping module resort to an interactive mode where a user can see scraped results immediately and adjust the module to resolve any scrapping errors on the fly.

Noting that only the back-end of the system is described here. To deploy this system, a suitable interface needs to be developed so that even inexperienced users without programming background can use the system. The system should also be able to record each decision made at each step by the user, so that the information can be used to improve the system. Moreover, the system should provide a mechanism to save and load a model, making the model reusable and supporting later refinement of the model.

5.5 Evaluation

We conducted an experiment based on the motivation underlying our framework, in which we want to help novices to build models with the similar quality as experts. In this experiment, we use the nuclear power domain, in which we know how the experts built the model and which features they use [1]. We constructed a dataset containing fourteen features as described in the paper, denoted by D_{exp}, and constructed another dataset containing top fourteen features recommended by our framework, denoted by D_{ont}. Note that six out of fourteen features recommended by our framework are similar to the features used by the experts.

Both datasets were split into training sets and test sets (30% of the whole datasets) such that both training sets contain an identical set of countries and both testing sets also contain an identical set of countries. Linear models were built from each dataset using standard linear regression method. In order to observe the overfitting phenomenon, a model was built by using only one attribute first, then the number of attributes was incremented to increase the complexity of the model

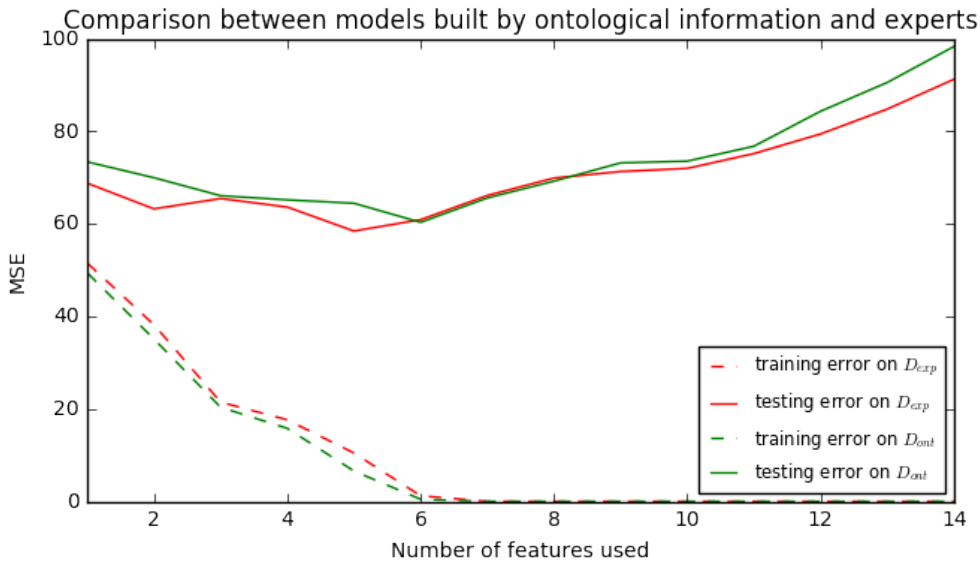

Figure 5.6: We compared models built by our framework *ont* with models built by experts *exp*. Linear models with different number of features were built using data of features recommended by our framework and features selected by experts. Average MSE values from 10-fold cross validation when testing these models on training and test sets are shown for each number of features used. The models built from our framework perform comparable to the models built by experts

being built. Different schemes were used to add attributes from each dataset to the model. For D_{ont}, each attribute was added in order with respect to the ranking of its corresponding ontological concept. For D_{exp}, each attribute was added in order corresponding to the ranking suggested by the experts. For each number of features used, we did train-test split for 50 times and the average mean square errors (MSEs) on corresponding training and test sets were computed. Note that the standard error of the MSEs is less than 0.1 when splitting the datasets 50 times. The results of this experiment are shown in Figure 5.6

From Figure 5.6, we can see that as the number of attributes used to build the models increase, the training errors decrease, while the testing errors increases. This phenomenon is the sign of overfitting, as the complexity of the models increase; we can expect their generalizations decrease. Also, we found that the errors obtained from the models built by the recommended features from our framework are comparable to the errors obtained from the models built by experts' features, which supports our claim. These results point out that the category organization in DBpedia pro-

vides useful and sufficient information in selecting features (attributes of a country) to build a good model. They also demonstrate the effectiveness of our feature recommendation procedure that makes use of the intuition underlying the conceptual distance in exploring the category organization and retrieving a set of relevant ontological concepts from this structure.

5.6 Summary

This chapter describes a model building framework that automatically constructs a model using relevant ontological concepts and data corresponding to those concepts. The data used in learning are selected by exploiting high-level knowledge separate from correlations within the data itself. We implemented this framework with DBpedia and Wikipedia to evaluate our approach. The prediction performance of a model built from the framework is comparable to a model built by the experts, which indicates the effectiveness of our approach in retrieving sufficient information from the ontology to build a good model. The implementation also helped build the model with very little human involvement.

6. CONCLUSIONS AND FUTURE WORK

This dissertation has attempted to address the changing needs of data analytics and data science techniques by re-examining the entire model building process not merely at the learning step, which is just one part of this process, and incorporating background knowledge along with rich semantic information expressed in an ontological structure into the process. This semantic information has been shown in our small scale studies in Chapter 4 and 5, to help prevent data from being used indiscriminately. The approach presented in this work not only opens up new opportunities for others by making it easier to produce more accurate models but also provides a new way to make sense of or interpret large amounts of data. Thus, if the web has played its part in the current inundation of data, then it may be said that the semantic web is helping to tame the deluge of data.

6.1 Significant Contributions

This section presents the significant theoretical and practical contributions of this work. The theoretical contributions include: establishing the connection between different notions of relevance at the conceptual and data levels, introducing a new way to assess input features using only their labels and certain structural information obtained from an ontology, and a new feature selection method that takes background knowledge and feature semantics into account in evaluating the relationship between features. Among the practical contributions of this work, we highlight two: the development of reusable SPARQL queries to effectively compute the conceptual distance from the ontology, and an end-to-end model building framework, which enables an inexperienced user to produce models with less effort and with similar performance to that of an expert. Experiments have provided evidence to support claims of the effectiveness and usefulness of the approach.

6.1.1 Theoretical Contributions

Each theoretical contribution is discussed in context of the research results, providing where appropriate the broader impacts for the machine learning communities.

6.1.1.1 A Connection between the Notions of Relevance at the Conceptual and the Data Levels

This work establishes a connection between the notions of relevance at the conceptual and the data levels. By connecting these two levels, the information obtained at the conceptual level can be used to inform whether or not the relationship between the entities at the data level is genuine. That is the notion of relevance at the conceptual level can be used as a means of extra discretion in preventing the indiscriminate use of the data. The effectiveness of the feature selection and feature recommendation, using the information at the conceptual level to select and recommend features, supports that the nexus formed between notions of relevance at the two levels is being correct.

6.1.1.2 Novel Means for Evaluating Relationship between Features

The notion of relevance at the conceptual level is realized via a metric, called conceptual distance, computed from structural knowledge, as codified in an ontology. This measurement becomes a fundamental tool in this work in evaluating relationship between features and incorporating semantic information into the model building process. Despite using only labels of features, the experimental results show that evaluating an input feature at the conceptual level via this metric performs comparably to the traditional data-driven measurements, and even better in some cases. These results are the evidences of the informativeness of the metric, indicating that it captures useful information from an ontology in assessing relationship between features. This metric is shown to be accessible across ontologies through simple and reusable SPARQL queries.

6.1.1.3 A New Feature Selection Method

The feature selection method is developed, using the criteria for evaluating feature redundancy and feature relevance defined by the conceptual distance. Experimental results obtained from applying this feature selection method on two real world datasets demonstrate the effectiveness and informativeness of the conceptual distance in selecting good feature subsets to build models with good performance, despite using only labels of features. This supports the usefulness of the metric and the notion of relevance at the conceptual level in addressing some classical machine learning problems.

6.1.2 Practical Contributions

Each practical contribution is discussed with in context of the research results, pointing to the possible broader impacts for the machine learning communities.

6.1.2.1 SPARQL Queries to Compute the Conceptual Distance

The SPARQL queries for computing the conceptual distance are developed in order to avoid loading the entire ontology or performing any pre-processing steps on a local machine. This approach is suitable when local resources are limited. The informativeness and effectiveness of the metric when applying in the feature selection method is an evidence showing the effectiveness of these queries in computing the metric from different ontologies. The queries are shown to be reusable across ontologies with only few changes.

6.1.2.2 An End-to-End Model Building Process

An end-to-end model building process is introduced, using the conceptual distance as a guideline for automatically exploring ontological structure and recommending relevant features for building a model. This process is designed such that it requires minimum human input, making it easy for inexperienced users to produce a model. An instance of this process was implemented with DBpedia and Wikipedia. Testing on this implementation demonstrates that semi-automated examination developed in our process can help novices to generate models with similar performance to that of a domain expert. This result further indicates that the feature recommendation procedure derived from the conceptual distance provides useful information to produce a good model.

6.2 Future Work

There are both immediate and long-term research goals for which this dissertation can be extended toward applying semantic information in addressing classical machine learning problems. The immediate research goals proposed here suggest additional evaluations of our approach to better understand the connection between the conceptual level and the data level. Long-term research

goals focus on exploring potential machine learning problems that require help from semantic information.

6.2.1 Immediate Extensions of the Research

Future research may be able to pinpoint crucial ontological structures so that the conceptual distance can be updated accordingly. This may make it possible to help better understand the connection between the conceptual level and the data level. So far, only information obtained at the conceptual level was used in feature selection. Combining information from both ontology and data may enable us to refine the criterion for feature selection and/or model selection.

6.2.2 Long-Term Future Research Goals

Semantic information may help in addressing classical machine learning problems other than feature selection. One of these problems includes how to appropriately handle missing data. Data semantics could determine a suitable value to replace the missing values. Model validation may be done by assessing whether the model built from the data is in an agreement with the semantic information and knowledge in an ontology. Performing such a validation could engender greater confidence and trust for those using a model that was built by others. Knowledge about a particular example obtained from an ontology could be useful in evaluating whether or not the relationship between two features is valid for that example. For instance, the correlation between cheese consumption and golf courses' revenue could be genuine in some countries. The most challenging question is perhaps how to use semantic information and knowledge in an ontology to generate natural language explanations or implications underlying a model [89]. These are the potential directions for long-term research that might be addressed in order to improve data analytics and data science techniques.

REFERENCES

[1] P. Nelson and C. M. Sprecher, "What Determines the Extent of National Reliance on Civil Nuclear Power?," in *Proceedings of the 49th Annual Meeting of the Institute of Nuclear Materials Management*, (Deerfield), pp. 72–79, Institute of Nuclear Materials Management, July 2008.

[2] W. Raghupathi and V. Raghupathi, "Big Data Analytics in Healthcare: Promise and Potential," *Health Information Science and Systems*, vol. 2, no. 3, 2014.

[3] R. Kohavi, N. J. Rothleder, and E. Simoudis, "Emerging Trends in Business Analytics," *Communications of the ACM*, vol. 45, no. 8, pp. 45–48, 2002.

[4] C. K. Leung and K. W. Joseph, "Sports Data Mining: Predicting Results for the College Football Games," *Procedia Computer Science*, vol. 35, pp. 710–719, 2014.

[5] V. Dhar, "Data Science and Prediction," *Communication of the ACM*, vol. 56, no. 12, pp. 64–73, 2013.

[6] A. Halevy, P. Norvig, and F. Pereira, "The Unreasonable Effectiveness of Data," *IEEE Intelligent Systems*, vol. 24, no. 2, pp. 8–12, 2009.

[7] C. Anderson, "The End of Theory: The Data Deluge Makes the Scientific Method Obsolete." `https://www.wired.com/2008/06/pb-theory/`, 2008. Accessed: 2019-10-02.

[8] G. Smith and E. Shah, "Data Dredging, Bias, and Confounding: They can All get You into BMJ and the Friday Papers," *BMJ*, vol. 325, no. 7378, pp. 1437–1438, 2002.

[9] J. Tukey, *Exploratory Data Analysis*. Addison-Wesley, 1977.

[10] N. Silver, *The Signal and the Noise: Why So Many Predictions Fail? but Some Don't*. Penguin Books, 2012.

[11] E. Dougherty, "Feature-Selection Overfitting with Small-Sample Classifier Design," *IEEE Intelligent Systems*, vol. 20, no. 6, 2005.

[12] K. Janowicz, F. van Harmelen, J. A. Hendler, and P. Hitzler, "Why the Data Train Needs Semantic Rails," *AI Magazine*, vol. 36, no. 1, pp. 5–14, 2015.

[13] T. Gruber, "Toward Principles for the Design of Ontologies Used for Knowledge Sharing," *International Journal of Human-Computer Studies*, vol. 43, no. 5–6, 1995.

[14] D. Koller and M. Sahami, "Toward optimal feature selection," *International Conference on Machine Learning*, pp. 284–292, 1996.

[15] A. L. Blum and P. Langley, "Selection of Relevant Features and Examples in Machine Learning," *Artificial Intelligence*, vol. 97, no. 1, pp. 245–271, 1997.

[16] I. Guyon and A. Elisseeff, "An Introduction to Variable and Feature Selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[17] R. Kohavi and G. H. John, "Wrappers for Feature Subset Selection," *Artificial Intelligence*, pp. 273–324, 1997.

[18] T. M. Mitchell, *Machine Learning*. McGraw-Hill, 1997.

[19] A. J. Miller, *Subset Selection in Regression*. Chapman & Hall/CRC, 2nd ed., 2002.

[20] E. Alpaydin, *Introduction to Machine Learning (Adaptive Computation and Machine Learning series)*. MIT press, 2nd ed., 2009.

[21] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer, 2001.

[22] K. P. Burnham and D. R. Anderson, *Model Selection and Multimodel Inference*. Springer-Verlag, 2002.

[23] S. J. Russell and B. N. Grosof, "Declarative Bias: An Overview," in *Change of Representation and Inductive Bias* (D. P. Benjamin, ed.), pp. 267–308, Boston, MA: Springer US, 1990.

[24] L. Todorovski and S. Dzeroski, "Declarative Bias in Equation Discovery," in *The 14th International Conference on Machine Learning*, pp. 376–384, 1997.

[25] J. Shavlik and G. Towell, "An Approach to Combining Explanation-Based and Neural Learning Algorithms," *Connection Science*, vol. 1, no. 3, pp. 233–255, 1989.

[26] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan-Kaufmann, 1988.

[27] J. Pearl, *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2009.

[28] D. Kaplan, *Structural Equation Modeling: Foundations and Extensions*. SAGE, 2008.

[29] T. Helleputte and P. Dupont, "Feature Selection by Transfer Learning with Linear Regularized Models," in *the European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 533–547, 2009.

[30] Z. Zhao, J. Wang, H. Liu, J. Ye, and Y. Chang, "Identifying Biologically Relevant Genes via Multiple Heterogenous Data Sources," in *The 14th International Conference on Knowledge Discovery and Data Mining*, pp. 839–847, 2008.

[31] A. B. Brahim and M. Limam, "New Prior Knowledge Based Extensions for Stable Feature Selection," in *The 6th International Conference of Soft Computing and Pattern Recognition*, pp. 306–311, 2014.

[32] W. Groves, "Using Domain Knowledge to Systematically Guide Feature Selection," in *The 23rd International Joint Conference on Artificial Intelligence*, pp. 3215–3216, 2013.

[33] V. C. S. Lee, D. Vickrey, and D. Koller, "Learning a Meta-Level Prior for Feature Relevance from Multiple Related Tasks," in *The 24th International Conference on Machine Learning*, pp. 489–496, 2007.

[34] O. Etzioni, M. Banko, S. Soderland, and D. Weld, "Open Information Extraction from the Web," *Communications of the ACM*, vol. 51, no. 12, pp. 68–74, 2008.

[35] E. Agichtein and L. Gravano, "Snowball: Extracting Relations from Large Plain-Text Collections," in *Proceedings of the 5th ACM International Conference on Digital Libraries*, pp. 85–94, 2000.

[36] R. J. Mooney and R. Bunescu, "Mining Knowledge from Text Using Information Extraction," *SIGKDD Explorations*, vol. 7, no. 1, pp. 3–10, 2005.

[37] A. Maedche and S. Staab, "Learning Ontologies for the Semantic Web," in *Semantic Web Workshop*, 2001.

[38] W. Wong, W. Liu, and M. Bennamoun, "Ontology Learning from Text: A Look back and into the Future," *ACM Computing Surveys*, vol. 44, no. 4, pp. 1–36, 2012.

[39] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann, "Dbpedia – a Crystallization Point for the Web of Data," *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 7, no. 3, pp. 154–165, 2009.

[40] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: A Core of Semantic Knowledge," in *The 16th International World Wide Web Conference*, pp. 697–706, 2007.

[41] G. A. Miller, "WordNet: A Lexical Database for English," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[42] D. Lenat, M. Prakash, and M. Shepherd, "CYC: Using Common Sense Knowledge to Overcome Brittleness and Knowledge Acquistion Bottlenecks," *AI Magazine*, vol. 6, no. 4, pp. 65–85, 1986.

[43] R. Hoehndorf, P. N. Schofield, and G. V. Gkoutos, "The Role of Ontologies in Biological and Biomedical Research: A Functional Perspective," *Briefings in Bioinformatics*, vol. 16, no. 6, pp. 1069–1080, 2015.

[44] B. Percha, Y. Garten, and R. B. Altman, "Discovery and Explanation of Drug-Drug Interactions via Text Mining," in *Pacific Symposium Biocomputing*, pp. 410–421, 2012.

[45] R. Hoehndorf, P. N. Schofield, and G. V. Gkoutos, "An Integrative, Transaltional Approach to Understanding Rare and Orphan Genetically Based Diseases," *Interface Focus*, vol. 3, 2013.

[46] P. H. Guzzi, M. Mina, and C. Guerra, "Semantic Similarity Analysis of Protein Data: Assessment with Biological Features and Issues," *Briefings in Bioinformatics*, vol. 13, pp. 569–585, 2011.

[47] S. Kohler, M. H. Schulz, P. Krawitz, S. Bauer, S. Dolken, C. E. Ott, C. Mundlos, D. Horn, S. Mundlos, and P. N. Robinson, "Clinical Diagnostics in Human Genetics with Semantic Similarity Searches in Ontologies," *American Journal of Human Genetics*, vol. 85, no. 4, pp. 457–464, 2009.

[48] F. T. Fonseca, M. J. Egenhofer, P. Agouris, and G. Camara, "Using Ontologies for Integrated Geographic Information Systems," *Transactions in GIS*, vol. 6, no. 3, pp. 231–257, 2002.

[49] H. Hwang, S. Shin, and C. Kim, "A User-Oriented GIS Search Service Using Ontology in Location-Based Services," in *The International Conference on Hybrid Information Technology*, pp. 209–218, 2006.

[50] Y. Jeong and S. H. Myaeng, "Feature Selection Using a Semantic Hierarchy for Event Recognition and Type Classification," in *The 6th International Joint Conference on Natural Language Processing*, pp. 136–144, 2013.

[51] H. Paulheim, "Exploiting Linked Open Data as Background Knowledge in Data Mining," in *The International Workshop on Data Mining on Linked Data*, pp. 1–10, 2013.

[52] S. T. Dumais, "Latent Semantic Analysis," *Annual Review of Information Science and Technology*, vol. 38, pp. 188–230, 2005.

[53] Z. S. Harris, "Distributional Structure," *Word*, vol. 10, no. 2, pp. 146–162, 1954.

[54] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and Their Compositionality," in *NIPS*, pp. 3111–3119, 2013.

[55] Y. Li, L. Xu, F. Tian, L. Jiang, X. Zhong, and E. Chen, "Word Embedding Revisited: A New Representation Learning and Explicit Matrix Factorization Perspective," in *The 24th International Joint Conference on Artificial Intelligence*, pp. 3650–3656, 2015.

[56] V. Lampos, B. Zou, and I. J. Cox, "Enhancing Feature Selection Using Word Embeddings: The Case of Flu Serveillance," in *International World Wide Web Conference Committee*, pp. 695–704, 2017.

[57] S. Chua and N. Kulathuramaiyer, "Semantic Feature Selection Using Wordnet," in *International World Wide Web Conference Committee*, pp. 166–172, 2004.

[58] A. V. Aho, M. R. Garey, and J. D. Ullman, "The Transitive Reduction of a Directed Graph," *SIAM Journal on Computing*, vol. 1, no. 2, pp. 131–137, 1972.

[59] S. Harispe, S. Ranwez, S. Janaqi, and J. Montmain, *Semantic Similarity from Natural Language and Ontology Analysis*. Morgan & Claypool Publishers, 2015.

[60] P. Resnik, "Using Information Content to Evaluate Semantic Similarity in a Taxonomy," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pp. 448–453, 1995.

[61] B. Sheehan, A. Quigley, B. Gaudin, and S. Dobson, "A Relation Based Measure of Semantic Similarity for Gene Ontology Annotation," *BMC Bioinformatics*, vol. 9, 2008.

[62] Z. Zhou, Y. Wang, and J. Gu, "A New Model of Information Content for Semantic Similarity in WordNet," in *Proceedings of the 2nd International Conference on Future Generation Communication and Networking Symposia*, pp. 85–89, 2008.

[63] N. Seco, T. Veale, and J. Hayes, "An Intrinsic Information Content Metric for Semantic Similarity in WordNet," in *Proceedings of the 16th European Conference on Artifical Intelligence*, pp. 1–5, 2005.

[64] D. Sanchez, M. Batet, and D. Isern, "Ontology-Based Information Content Computation," *Knowledge Based Systems*, vol. 24, no. 2, 2011.

[65] Z. Wu and M. Palmer, "Verbs Semantics and Lexical Selection," in *Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics*, pp. 133–138, 1994.

[66] D. Lin, "An Information-Theoretic Definition of Similarity," in *Proceedings of the the 15th International Conference on Machine Learning*, pp. 296–304, 1998.

[67] M. B. Aouicha, M. A. H. Taieb, and M. Ezzeddine, "Derivation of "is a" Taxonomy from Wikipedia Category Graph," *Engineering Applications of Artificial Intelligence*, vol. 50, pp. 265–286, 2016.

[68] H. Liu and H. Motoda, *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer Academic Publishers, 1998.

[69] H. Almuallim and T. G. Dietterich, "Learning with Many Irrelevant Features," in *Proceedings AAAI-91*, pp. 547–552, 1991.

[70] K. Kira and L. A. Rendell, "The Feature Selection Problem: Traditional Methods and a New Algorithm," in *Proceedings AAAI-92*, pp. 129–134, 1992.

[71] J. R. Quinlan, "Induction of Decision Trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.

[72] J. R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[73] R. Tibshirani, "Regression Shrinkage and Selection via the Lasso," *Journal of the Royal Statistical Society*, vol. 58, no. 1, pp. 267–288, 1996.

[74] A. E. Hoerl and R. W. Kennard, "Ridge Regression: Biased Estimation for Nonorthogonal Problems," *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[75] L. Yu and H. Liu, "Efficient feature selection via analysis of relevance and redundancy," *Journal of Machine Learning Research*, vol. 5, pp. 1205–1224, 2004.

[76] M. A. Hall, "Correlation-Based Feature Selection for Discrete and Numeric Class Machine Learning," in *Proceedings of the 17th International Conference on Machine Learning*, pp. 359–366, 2000.

[77] G. H. John, R. Kohavi, and K. Pfleger, "Irrelevant Feature and the Subset Selection Problem," in *Proceedings of the 11th International Conference on Machine Learning*, pp. 121–129, 1994.

[78] H. C. Pang, F. Long, and C. Ding, "Feature Selection Based on Mutual Information: Criteria of Max-Dependence, Max-Relevance, and Min-Redundancy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1226–1238, 2005.

[79] P. Mitra, C. A. Murthy, and S. K. Pal, "Unsupervised Feature Selection Using Feature Similarity," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 301–312, 2002.

[80] D. A. Bell and H. Wang, "A Formalism for Relevance and Its Application in Feature Subset Selection," *Artificial Intelligence*, vol. 41, no. 2, pp. 175–195, 2000.

[81] G. Zhu and C. A. Iglesias, "Computing Semantic Similarity of Concepts in Knowledge Graphs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 1, pp. 72–85, 2017.

[82] M. C. Monuteaux, L. K. Lee, D. Hemenway, R. Mannix, and E. W. Fleegler, "Firearm Ownership and Violent Crime in the U.S.," *American Journal of Preventive Medicine*, vol. 49, no. 2, pp. 207–214, 2015.

[83] C. D. Manning, P. Raghavan, and H. Schuetze, *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[84] S.Janpuangtong and D. A. Shell, "Helping Novices Avoid the Hazards of Data: Leveraging Ontologies to Improve Model Generalization Automatically with Online Data Sources," *AI Magazine*, vol. 37, no. 2, pp. 19–32, 2016.

[85] J. M. Kleinberg, "Authoritative Sources in a Hyperlinked Environment," *Journal of the ACM*, vol. 46, no. 5, pp. 604–632, 1999.

[86] C. S. Bhagavatula, T. Noraset, and D. Downey, "Methods for Exploring and Mining Tables on Wikipedia," in *Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics*, (New York), pp. 18–26, ACM, 2013.

[87] G. Pirro, "REWOrD: Semantic Relatedness in the Web of Data," in *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, (Palo Alto), pp. 129–135, AAAI Press, 2012.

[88] A. Freitas, J. G. Oliveira, S. O'Riain, E. Curry, and J. C. P. D. Silva, "Querying Linked Data Using Semantic Relatedness: A Vocabulary Independent Approach," in *Proceedings of the 16th International Conference on Applications of Natural Language to Information Systems*, Lecture Notes in Computer Science, (Berlin), pp. 40–51, Springer, 2011.

[89] G. Shumeli, "To Explain or to Predict," *Statistical Science*, vol. 25, no. 3, pp. 289–310, 2010.